



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

Masters' Degree in Informatics Engineering
Dissertation

SUnSET

On the suitability of USDL for service description-A case study of a
telecom operator

September 4, 2013

Filipe João Vinagre Barata
fbarata@student.dei.uc.pt

Advisors at DEI:
Prof. Dr. Jorge Cardoso

Advisor at SAPO:
Eng. António Cruz



Abstract

During the last decades there has been tremendous growth in the services sector. This event was due to the great technological advancement that was given in communications and devices where the Internet was one of the reasons for this to happen. Through this, both consumers and companies managed a way to be always connected to any place and person. This led to an increasingly appearance of different services, such as Cloud services. Nowadays, companies have turned to the Cloud to reduce costs by outsourcing their non-core competencies or betting on new differentiating services. The Cloud has allowed the emergence of new companies, increasing quality of services provided as well as allowing greater competition. However, this increase of services has led to it being too complicated to find and compare. To solve this problem it is necessary to be able to describe the service and its features, both in a functional and non-functional way. This means that languages such as WSDL are not sufficient, because they are not able to capture and describe a complete service offer. It is crucial to have a language that can capture these aspects so important to the consumer. As such, we used USDL to describe the services from the business, operational and technical perspective. After we described this services they are ready to be on a common place and with easy access to the consumer. In order to outline this problem services marketplace were created so that all of them can be on a single distribution channel. Nowadays, service marketplaces are increasingly wherein each marketplace has its own way of describing services. The lack of use non-standard languages or with a large degree of dissemination for services description reduced the ability of marketplace interoperability with other internal and external information systems. With this thesis we propose to provide methods and tools to do a Export/Import application for services described in SAPO proprietary data scheme into USDL as well as the reverse actions. This document is the final report that will be presented in September 2013.

Keywords: Service economy, Cloud services, Service description, Service marketplaces, USDL, Linked-USDL, SDB data scheme

Contents

Abstract	ii
List of Tables	viii
List of Figures	x
List of Acronyms	xii
1 Introduction	1
1.1 Background	1
1.2 Motivation and Problem Description	4
1.2.1 Motivation	4
1.2.2 Problem Description	6
1.3 Objectives and Requirements	6
1.4 Challenges	8
1.5 Approaches	9
1.5.1 Methods	9
1.6 Sheduling	10
1.7 Risk Management	14
2 Related Work	17
2.1 Service Based-Economy	17
2.1.1 How does the Internet influences the economy?	18
2.1.2 ICT and Services Employment Impact in Economy	20
2.1.3 Services	21
2.1.4 Cloud Services	23
2.2 Service-Oriented Architecture (SOA)	26
2.3 Service Descriptions	28
2.3.1 Web Services Description Language (WSDL)	28
2.3.2 Unified Service Description Language (USDL)	28
2.3.3 Semantic Approaches	30
2.4 Service Marketplaces	33

2.4.1	SAPO Services Marketplace and their connection with SDB	38
3	Service Description Models	41
3.1	SDB data scheme	41
3.2	USDL Model	43
3.2.1	Business	43
3.2.2	Technical	46
3.2.3	Operational	46
3.3	Linked-USDL Model	47
3.4	Mismatch	49
3.4.1	Mismatch Conclusion	53
4	Analysis and Specification	55
4.1	Requirements Analysis	55
4.1.1	Requirements List	55
4.1.2	Use Cases	56
4.1.3	Interface Mockups	56
4.2	SUnSET Architecture	58
4.2.1	Export Engine	61
4.2.2	Search Engine	62
4.2.3	Web Interface	62
5	Semantic Models	63
5.1	Service Lifecycle Ontology	63
5.1.1	Motivation	63
5.1.2	Challenges	63
5.1.3	Model	64
5.2	SAPO Services Taxonomy	69
6	Service Model Mapping	71
6.1	Approaches to Mapping	71
6.2	Establishing Mappings	72
6.2.1	General Module Information	72
6.2.2	Module Dependencies	74
6.2.3	SDB data scheme extension: Model	74
6.3	Export/Import Application	84
6.3.1	System technologies	84
6.3.2	Implementation	84
6.3.3	Configurations	85

CONTENTS

6.4	Evaluation	87
6.4.1	Test Plan	87
6.4.2	Test Case	88
6.4.3	Results	89
7	Conclusion	91
7.1	Summary	91
7.2	Future Work	92
	Appendices	96
A	GIS Use Case	99
A.1	Introduction	99
A.2	Why GIS?	99
A.3	Technologies Used	100
A.4	Methodology	100
A.5	Linked-USDL Service Modeling	101
A.5.1	Vocabulary Prefixes	101
A.5.2	Service Instance	102
A.5.3	Legal	103
A.5.4	SLA	103
A.5.5	Pricing	103
A.5.6	Service Vocabulary	109
A.5.7	General Features	110
A.5.8	Advantages	111
A.5.9	Security Features	111
A.5.10	Customer Support	112
A.5.11	Service Requirements	112
A.5.12	Overview of GIS Use Case	112
	Appendices	99
B	Requirements Lists	115
B.1	Functional Requirements	115
B.2	Non-functional Requirements	115
B.2.1	Usability Requirement List	116
B.2.2	Reliability Requirement List	116
B.2.3	Design Requirement List	117
B.2.4	Implementation Requirement List	117
B.2.5	Interface Requirement List	117

C Service Lifecycle Ontology - Turtle Schema	119
D SAPO Services Taxonomy - Turtle Schema	125
D.1 Top Level Concepts	125
D.2 Property	125
D.2.1 Functional Property	126
D.2.2 Non-Functional Property	126
D.2.3 Interface	126
D.2.4 Support Properties	126

List of Tables

1.1	Risk Management	15
3.1	Differences between schemes	51
3.2	Number of elements in each domain available in these service description languages	52
5.1	Service Lifecycle Ontology Classes	64
5.2	Service Lifecycle Ontology Properties	66
6.1	overview of concepts used in SDB data scheme extension	74
6.2	ResourceDescription concept	75
6.3	DevelopmentCycle concept	75
6.4	WorkProfile concept	76
6.5	Area concept	76
6.6	EntityAttribute concept	77
6.7	Project concept	77
6.8	Schema concept	78
6.9	Specification concept	79
6.10	Operation concept	80
6.11	SecurityAttributeExt concept	81
6.12	Container concept	82
6.13	EntityAttributesType concept	82
6.14	SpecificationType concept	83
6.15	SecurityAttributeExtType concept	83
6.16	Functional Tests list	89
6.17	Usability Tests list	89
6.18	Reliability Tests list	90
6.19	Design Tests list	90
A.1	GIS Use Case and SDB data schema differences	112
B.1	Functional Requirements list	115
B.2	Usability Requirements list	116
B.3	Reliability Requirements list	116

LIST OF TABLES

B.4 Design Requirements list 117
B.5 Implementation Requirements list 117
B.6 Interface Requirements list 117

List of Figures

1.1	Project Intermediate Planning (October 2012)	11
1.2	Project Intermediate Planning (January 2013)	12
1.3	Final Project Planning (August 2013)	13
2.1	Wireless Internet access overtaking broadband subscriptions, in OECD Internet Economy Outlook 2012	18
2.2	Business with a broadband connection in 2011, in OECD Internet Economy Outlook 2012	20
2.3	Share of ICT employment in business sector employment, 1995 and 2009, in [41]	22
2.4	ICT sector employment in the OECD area by sector, 1995-2009, in [41]	23
2.5	SaaS Multilayer Model [17]	26
2.6	The α -USDL Model [12]	29
2.7	USDL Version 3.0 M5	29
2.8	WSDL versus USDL [12]	30
2.9	OWL-S service description elements [35]	32
2.10	Services Description Language Chronology	32
2.11	SAPO Services Marketplace	34
2.12	Amazon Web Services Marketplace [35]	35
2.13	SAP Services Marketplace[35]	35
2.14	SDB Logical Architecture[4]	39
3.1	USDL-Core	48
3.2	USDL-Pricing	48
3.3	USDL-SLA	48
4.1	Use Case A - Export	57
4.2	Use Case B - Search	57
4.3	Mockup A - System Login	58
4.4	Mockup B - Export	59
4.5	Mockup C - Search	59
4.6	Software Life Cycle - How FMC fills the gap[30]	60

LIST OF FIGURES

4.7	SUnSET architecture	61
5.1	Service Lifecycle example	65
5.2	Service Lifecycle Ontology Diagram	68
6.1	Class diagram of the package that captures the SDB data scheme concepts.	73
A.1	GIS Studio Overview	100
A.2	GIS Studio Pricing	101

List of Acronyms

B2B Business to Business

ICT Information and Communication Technologies

SaaS Service as a Software

XML Extensible Markup Language

RDF Resource Description Framework

IoS Internet of Services

OWL-S Semantic Markup for Web Services

SOA Service Oriented Architecture

USDL Unified Service Description Language

WSDL Web Service Description Language

SAWSDL Semantic Annotations for WSDL

WSMO Web Service Modelling Ontology

WSML Web Service Modelling Language

OECD Organization for Economic Co-operation and Development

1

Introduction

This chapter gives an introduction to some of the topics addressed in this thesis and it is organized in seven sections. Section 1.1 describes the background of the main topics discussed through the project, e.g, Service Description, Cloud computing, Marketplaces, etc. Section 1.2 explains the motivation and relevance of the current work in the fields as well as the problem description of the project presented throughout this thesis. Section 1.3 explains the overall goals and requirements of the work proposed, such as, the analysis of the different schemas or establishing mappings between these schemas. Section 1.4 presents the challenges to fully describe software services. Section 1.5 explains the approaches taken to achieve the above described goals, e.g, methods and tools. Section 1.6 gives an overview of the preliminary thesis planning. The last Section 1.7, shows the potential risks that can be found throughout the project.

1.1 Background

In the last decades there as been a change in the world economy due to the globalization and the exponentially growth in information and communication technologies (ICT), in the so called industrialized economies. This is mainly due to the fast technology progress in the ICT sector and the continuously falling prices of all devices [19]. Accordingly to the authors in [19] “ICTs constitute not only infrastructure and hardware but also data, information and knowledge. Infrastructure without the necessary skill to convert data to information and information to knowledge and incorporate its use into social and economic processes is not ICT”. For example, ICT can be used to access information from the web, to facilitate communication between entities, store information, or, as said in [39], “enables firms to restructure their

CHAPTER 1. INTRODUCTION

organizations(...), to re-engineer business processes(...), and to develop completely new products”. There is a wilderness of possibilities. As outcome, a large part of the economy growth is due to the “innovative applications in manufacturing” [50] like the Manufacturing Requirements Planning (MRP) in the 1970s and Enterprise Resource Planning (ERP) in the 1990s, and mostly because of the service sector, turning this one of the biggest and fastest-growing in the world [40]. Knowing this, we need to understand what is a service.

The best way to begin the service explanation is with a definition. Accordingly to Cardoso et al. [14]:

“In business, a service is the non-material equivalent of a good. It is considered to be an activity witch is intangible by nature witch is provided by a service provider to a service consumer to create a value possibly for both parties.”

Probably most of us have already used services without knowing what they are. As an example, the user asks for a translation online or accesses an online chat in order to get help. Almost everything nowadays is a service, so we have to be very careful when we try to mention something as a service. We can include services in Business services, e-Services and Web services, which it will be discussed with more detail in Section 2.1.3.

Nowadays, most of us have several devices, like computers, tablets, phones, and a continuous wireless or 3G connection access, and so we are always connected to information. This permanent access to those devices allowed companies to transfer their applications and data to remote data centers, and “installed in the compute cloud” [26]. This allowed users to access all of this, in the Cloud and on demand, wherever they are [11], bringing a large number of advantages to the previous paradigm. With Cloud computing, users have no need to manage software installations and updates, or to change outdated hardware components. This, and the personnel costs are all processed by the requested Cloud company. With this in mind, “computing Clouds render users with services to access hardware, software and data resources” [53]. Depending on the type of provided capability, there are different categories of cloud services such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), Communication as a Service (CaaS) and so on. In our case we are more interested in SaaS. This is where a software or an application is hosted as a service and provided to the users across all the internet. Users do not need to locally run applications. As an example, we can use Google web-based office in alternative to the locally office applications. “Many products are now offered as services under the umbrella of cloud computing” witch brings us new challenges [51][38] as we describe below.

1.1. BACKGROUND

Services, such as SaaS, are becoming tremendous important in our society as they are increasing and used continuously. However, in order to find the required service to fulfill their needs, the consumer has typically the same problem. First, it is necessary to make a search in Google¹, so that the consumer may have a list of the available services. Next, the consumer has to see page by page, in order to take some notes about the service, so that in the end can have a list of every service. The last step, is to compare manually this list. It is not hard to see that this process is time consuming. But how can the consumer find and compare services in a fast and simple way? One answer for this are the electronic marketplaces. In the last years, marketplaces for products have gained a large market share, enabling consumers to buy products without great effort and in the comfort of home. One of the best known examples is Amazon marketplace², where consumers have a huge choice of products. The same is happening with services, as seen in SAPO Services³, Amazon Services⁴ or SAP Services⁵. The term Internet of Services⁶ (IoS) refers to the infrastructure that enable the supplying of universal services to consumers. “Services are seen as tradable goods that can be offered on service marketplaces by their providers to make them available for potential consumers”. But this only solves the finding problem. In order to take full advantage of a service marketplaces, the consumer must have a way to compare services. In order to do that, there must be a common language to describe services. Accordingly to IoS, services needed to be described in a “form to enable their publication, discovery, selection, contracting and monitoring”. This means that services must be described in a technical, business and operational perspective, witch could bring many advantages for both the consumer and the provider [15].

In order to describe services in this way, despite of their business area or domain, and to allow services to became tradable and consumable, the Unified Service Description Language⁷ (USDL) was created. Nevertheless, there are some limitations like complexity and extension problems. The first problem is that this specification view on services resulted in more than a hundred classes, taking the data model to an unwanted complexity level. Despite this complexity to describe a service, it was insufficient to describe all services domain. New elements necessary could only be implemented by the USDL Incubator

¹<http://www.google.com>

²<http://www.amazon.com>

³<https://store.services.sapo.pt/en/>

⁴<http://aws.amazon.com/>

⁵<https://websmp207.sap-ag.de/>

⁶<http://www.internet-of-services.com/>

⁷<http://www.internet-of-services.com/index.php?id=288&L=0>

Group⁸. In order to overcome those problems, a new model called Linked-USDL⁹ was created. It uses Semantic Web Technologies as its foundation, providing customizability, extensibility and interoperability. The complexity of the model description decreased due to a simplification made relatively to USDL. Using the Semantic Web Technologies, everyone can implement new elements, providing the means to describe specific domains. This already shows some advantages over the USDL model, but a more specific explanation will be addressed later in Section 2.3.

1.2 Motivation and Problem Description

1.2.1 Motivation

The motivation for conducting this thesis is based mainly on three aspects:

1. The need for a language that describes services in an uniform and standard way to achieve better acceptance by the market;
2. The need for a domain specific service language, e.g, it allows a particular type of service to be expressed more clearly than a domain generic language;
3. Global scale marketplace integration/interoperability to enable services exchange between marketplaces and other information systems.

1. *The need for a language that describes services in an uniform and standard way to achieve better acceptance by the market.* Implementing standards consistently provides benefits for companies, regardless of their size, business sector or which country they are based in. With a language that describes services in an uniform and standard way it is possible to make industry more efficient and making international trade more accessible. Accordingly to the authors in [25], it could bring a generous number of benefits such as: decreasing waste and internal costs, reducing risks, enabling international expansion and supporting development of new products and markets.

2. *The need for a domain specific service language, e.g, it allows a particular type of service to be expressed more clearly than a domain generic language.* A domain specific language allows that a service

⁸<http://www.w3.org/2005/Incubator/usdl/charter>

⁹<http://www.linked-usdl.org/>

1.2. MOTIVATION AND PROBLEM DESCRIPTION

can be expressed more clearly because of the expressive power and, therefore, it brings higher productivity and optimization as well making the language easier to use. There are many technical languages that could delivered this kind of expressiveness, however we need one language that could delivered that type of description with the addition of other domains in order to fully describe services, such as, the operational and business domain. Consequently, we will need just one language with a much simplified description process.

3. Global scale marketplace integration/interoperability to enable services exchange between marketplaces and other information systems. Once services are described, then they are suitable to be in a place to trade services. This happens, e.g, to products in Amazon marketplace. The intermediate mechanism between the consumer and the service will be the marketplace. It is through this mechanism that the consumer may demand for a particular service, and/or can set properties in his interest so that the Marketplace can respond with services that have these same properties. As result, the consumer can save a lot of time searching and comparing services. However, the advantage is not only for consumers but also for the provider. In this case, the possibility of having multiple services on a single site, brings many advantages. As example, the provider can have a unique distributing channel for all services. One advantage of this is in terms of cost, since the provider doesn't have to spend much money on advertising every single service [27]. The integration of different systems is a challenge that many companies are facing nowadays, and it is not difficult to see why. Different representations of the same domain make it difficult to automatically integrate data and services, turning this a time consuming and expensive job. As Soon-Yong Choi said [16]:

“The need for interoperability in technologies is evident if we are to facilitate transactions of goods and services that may involve firms and consumers in traditionally separated markets. In order to support production, trading, and consumption of these products and services in an integrated manner, computing and networking technologies must be interoperable with other products, Web pages, payment systems and user interfaces based on different computing platforms as well different needs and preferences of users.”

Nowadays, Extensible Markup Language (XML), a “markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable”, has become the standard for information exchange between systems. However, in order to have a global marketplace

integration/interoperability to enable services exchange, we need to have a common language with specific rules so that we can use across different systems.

1.2.2 Problem Description

Due to the huge competition and the incessant changing in the business world, it is necessary that the businesses can always focus on ways of differentiation. As such, to be competitive one must always be one step ahead of the market investing in various branches. If the market adopts a certain path they are prepared to follow it.

After the first meeting with SAPO¹⁰, a subsidiary company of Portugal Telecom (PT), it was presented the following problem: The SAPO developed in-house a marketplace of services based on a proprietary data scheme. The lack of use for non-standard languages or languages with a large degree of dissemination for services description reduced the ability of marketplace interoperability with other internal and external information systems. As such they need the marketplace of services to be prepared for a possible transition that may occur in the way that services are described. This means that they want their services described through a service description language that is uniform and standard.

1.3 Objectives and Requirements

In this project, we have one main objective that decomposes in three secondary objectives:

1. Develop an application that automatically does the export of Sapo services for a USDL description and also the export of services described in USDL to a proprietary Sapo services format. ;
 - (a) SDB data scheme, USDL and Linked-USDL analyse and their respective comparison;
 - (b) Establishing mappings between SDB data scheme and USDL;
 - (c) Implementation, testing and evaluation of the application.
2. Create a Linked-USDL extension to represent the lifecycle of a service.
3. Create a simple search engine to illustrate the use of Linked-USDL and Sapo services.

¹⁰<http://www.sapo.pt>

1.3. OBJECTIVES AND REQUIREMENTS

1. Develop an application that automatically does the export of Sapo services for a USDL description and also the export of services described in USDL to a proprietary Sapo services format.

The main objective of this project is to have one application that can automatically make the export of SDB data scheme into a USDL services description and/or between USDL services description into SDB data scheme. In order to achieve this main objective it is necessary to perform the following sub-objectives:

1a. SDB data scheme, USDL and Linked-USDL analyse and their respective comparison.

The first secondary objective is the need to verify the structural differences between the SDB data scheme, USDL and Linked-USDL. With the results of this analysis will be possible to realize the level of complexity required to describe a particular service. If necessary, it can be useful to make recommendations to the Telecommunications Operator in order to improve services description.

1b. Establishing mappings between SDB data scheme and USDL.

The next secondary objective consists on the process of establish mappings between the two schemas. We have to identify the mappings that have a direct connection and the ones that are out of context. For these last, it has to be proposed an extension, in order to do this mapping. Since the main target is to make the export between schemes direct and automatically, this is an essential part of the project. It must be created a set of rules that will allow a direct conversion through each mapping of each scheme. As stated earlier, it has to be Universal for any existing service, both in SAPO Services data schema as through its description in USDL.

1c. Implementation, testing and evaluation of the application.

The last secondary objective, is related to the creation, testing and evaluation of the application.

2. Create a Linked-USDL extension to represent the lifecycle of a service.

Linked-USDL is a language that is going through a redesign phase in all their modules. Furthermore, we saw that there are some areas that are not covered by this language. One of them is the service lifecycle in which we think it will add some value for service description.

3. Create a simple search engine to illustrate the use of Linked-USDL and Sapo services.

The idea behind this objective is demonstrate how we

can use Linked-USDL and a Taxonomy to describe some information presented in Sapo services marketplace. It is not our aim to model all these services but merely proving that Linked-USDL is able to describing services.

1.4 Challenges

In this early work were identified seven main challenges that we have to take into account:

1. Lack of SDB data scheme documentation;
2. Difficulty in having the desired SDB data scheme because of bureaucracy problems (pricing) and a lot of business related information is not yet available in SDB (SLA's, Service Level, etc.);
3. Different complexities and different technologies of the SDB data scheme, USDL and Linked-USDL;
4. Difficulty creating the correct mappings between schemes due to some orphan elements that can appear between these same schemes;
5. Possibility of creating an extension to mapping orphan elements;
6. Correct application functioning without information loses between the conversions.
7. Create a Linked-USDL extension to represent all the services lifecycles.

1. Lack of SDB data scheme documentation. The lack of available documentation results in much more time needed to study, analyze and understand the concept behind of SDB data scheme.

2. Difficulty having the desired SDB data scheme because of bureaucracy problems (pricing) and because a lot of business related information is not yet available in SDB (SLA's, Service Level). During the first semester was available a first version of SDB data scheme. This first version only describes services in a technical perspective being far from the desirable objective. Our intention was to have the services described in the business, operational and technical perspectives. However, some of those information is not yet available in the SDB platform or simply by bureaucratic issues were not available.

3. Different complexities and different technologies of the SDB data scheme, USDL and Linked-USDL. The fact of being new technologies, e.g, SDB data scheme and the different services description languages, they takes time to learn.

4. Difficulty creating the correct mappings between schemes due to some orphan elements that can appear between schemes. As both schemes were designed with different processes and perspectives in mind, and being one created specifically to describe services for the closed platform (SDB data scheme) and the other for describing services in general, it is possible that there may be orphan elements. These elements are those that are not represented in one of the schemes

5. Possibility of creating an extension to mapping orphan elements. SDB data scheme describes services in a more technical way. It is also focus on a lifecycle oriented way. As such, it will probably be necessary to create an extension in order to map those elements.

6. Correct application functioning without information loses between the conversions. Since for any final consumer, what matters most is to receive a product that works as it says. In this case in order to have a product that can work correctly it is necessary that the export could be performed without any information loss.

7. Create a Linked-USDL extension to represent all the services lifecycles. There is a slight complexity in defining an extension that could be valid for all the services.

1.5 Approaches

1.5.1 Methods

USDL and Linked-USDL will be used as service description languages in this project. USDL allows an unified description of business, operational and technical aspects of the service. It can describe various types of services ranging from manual to electronic services. Their main focus are on business aspects such as provisioning, composition, bundling, pricing and legal aspects among others. Linked-USDL is based on semantic web and linked data principles developed to achieve a wider global acceptance than his predecessor USDL [13]. The author defines the goal of Linked-USDL to "develop an ontology

CHAPTER 1. INTRODUCTION

to represent services by establishing explicit ontological links to other existing ontologies emerging from Linked Data initiatives.” As in USDL we can achieve a more complete description with more than just operational and technical perspectives. In fact it is intended to focus in the business aspect of the service, where the non-functional and functional elements allowed a better service description comparing to other languages, such as WSDL.

We will use USDL together with the SDB data scheme in the export process. Linked-USDL will be used for analysis and some service descriptions.

Programmatic Languages. The chosen language to make the export was C#. It was one of the project requirements along with webservice. It will be created a website with different technologies in order to provide the export process, such as, Active Server Pages(ASP), JavaScript(JS) and Asynchronous JavaScript and XML(AJAX).

Libraries There are also some libraries that would be used in this project. The first one is jQuery for simplifying some client-side scripting of HTML. The second one is the library DotNetRDF that is used for the search engine.

1.6 Sheduling

The project is being developed throughout the year of 2012/2013, in collaboration with SAPO and the Department of Informatics and Engineering of University of Coimbra. A status meeting is made every week with the advisor Professor Jorge Cardoso. When needed, there will be meetings with advisor Eng. António Cruz at SAPO Lisbon. Figure 1.1 and Figure 1.2 represent Gantt charts for the first and second semester respectively. The first one represent the expected of work distribution and the second represents the correction of the work plan which was made during the semester. Analysing the two project plans we can see that there are a few differences between them. It was introduced Linked-USDL in the implementation phase, as we can see in Figure 1.2 were is indicated the Linked-USDL study, but the main differences between the two planning were in the started and finished time for each task. Figure 1.3 represent the final project planning, with the respective changes. The second semester was focused more on the implementation and the creation of two Reports.

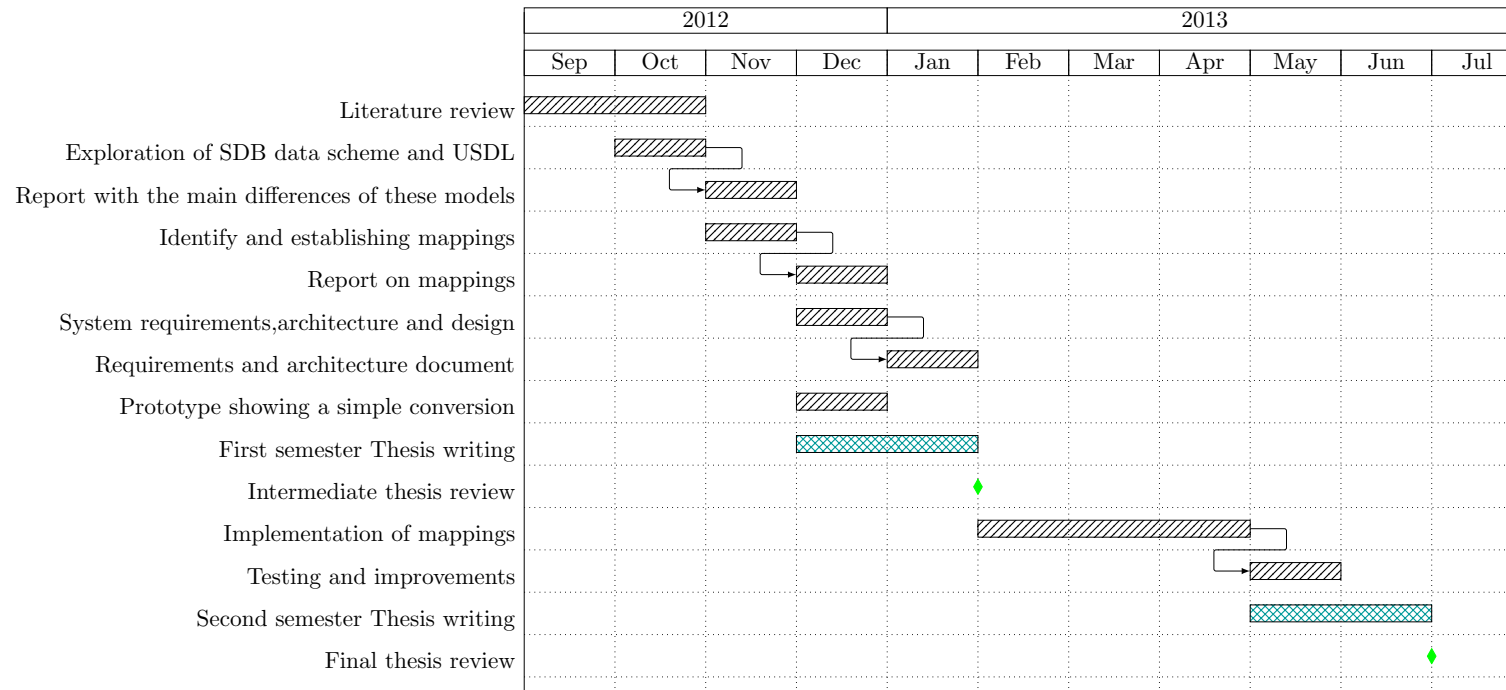


Figure 1.1: Project Intermediate Planning (October 2012)

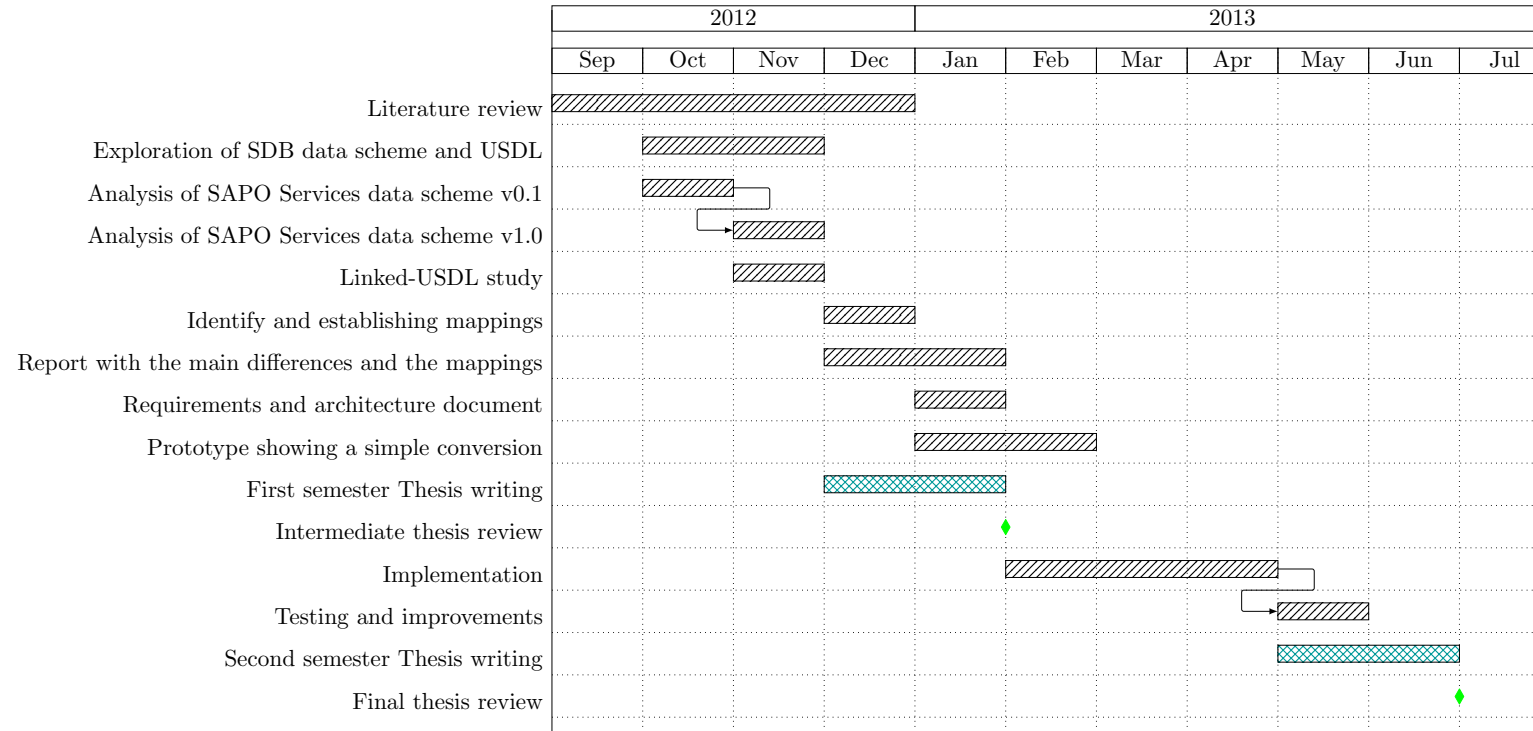


Figure 1.2: Project Intermediate Planning (January 2013)

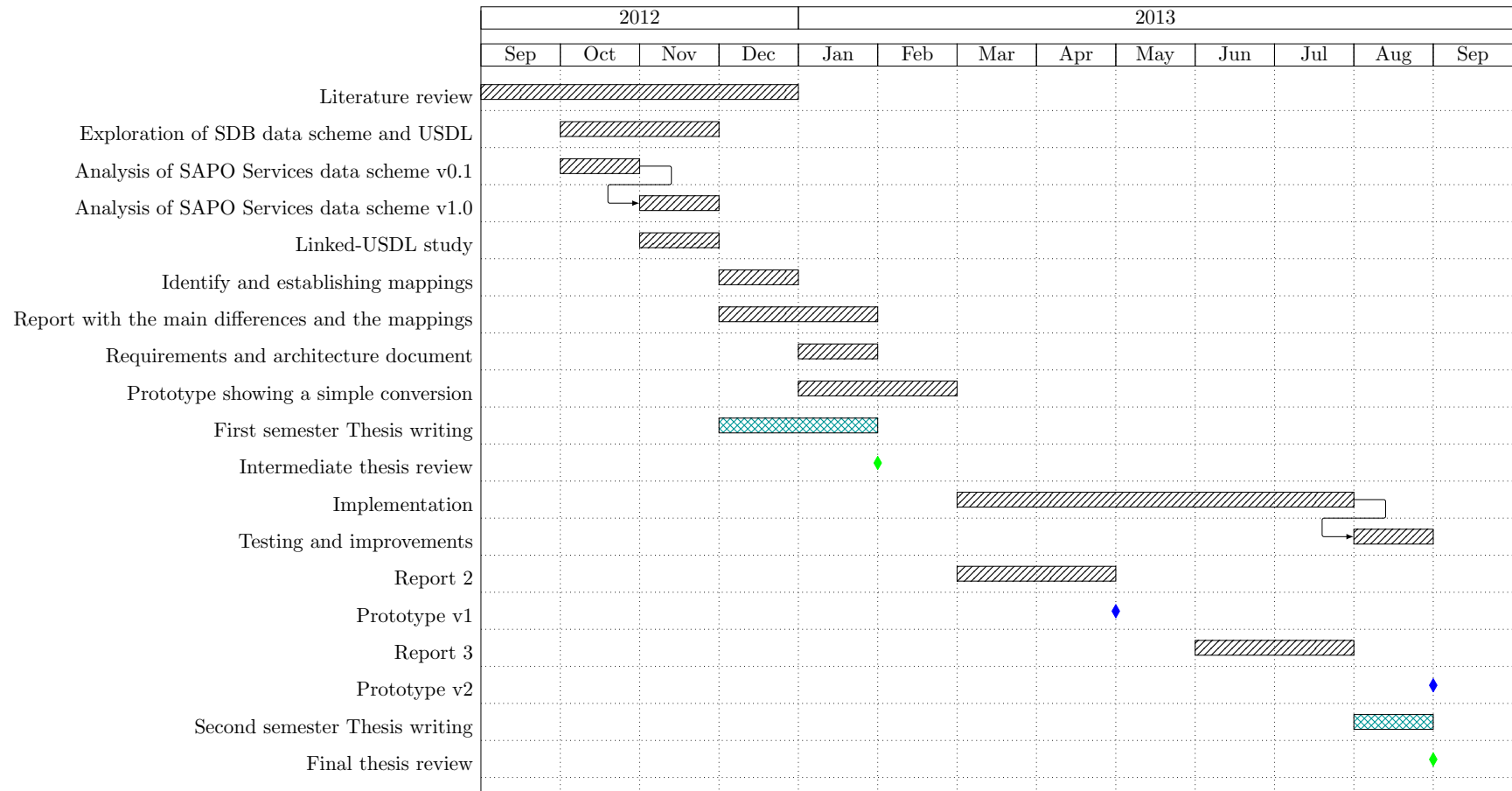


Figure 1.3: Final Project Planning (August 2013)

1.7 Risk Management

During all activities such as this project there are risks which we can find. They may appear in situations like legal liabilities, natural causes, financial problems, etc. We need to minimize or avoid these risks by doing a risk management. Accordingly to Sommerville [47] it is defined by these stages: identification, analysis, planning and monitoring of risks. In Table 1.7 we presented the risks analysis of this project.

Table 1.1: Risk Management

Risk	Affects	Type	Description	Probability	Impact	Strategy	Potencial Indicators
Timetable poor estimates	Project	Estimation	Projection error in time estimated for each task.	moderate	tolerable	Perform estimates based on previous work experiences.	Delay in project tasks
Advisors or feedback unavailable	Project	People	Advisors not available due to project external factors(business meetings or illness).	very low	serious	Schedule regular meetings with advisors in order to decrease issues during the course of the project. .	Advisors unreachability
Requirements change	Project	Requirements	Possible change(s) in the project requirements.	low	serious	Schedule meetings with project stakeholders in order to detect and discussed possible requirements change.	Misunderstandings in meetings
Implementation difficulties	Product	Technology	Student have difficulties with the technology or tools when implementing the system.	moderate	tolerable	Allocate more efforts and have alternatives when defining the system and technologies to be used.	Delay in specific feature implementation.
Underestimated implemented time	Product	Technology	Software implementation takes too much time due to the domain lack of knowledge.	moderate	tolerable	Defined a large block of time for the implementation.	Delay in specific feature implementation .
Student demotivation	Project and Product	People	The student finds no motivation to continue its project development.	very low	catastrophic	Talk urgently to stakeholders so that a solution to this lack of motivation can be found.	Tremendous delay in the execution of tasks and the overall project.
Incorrect mappings through the different schemes	Project and Product	Technology	Incorrect or incomplete mappings leads to a bad conversion between the different schemes (e.g, Linked-USDL and SAPO data scheme).	moderate	serious	Schedule more time for the implementation of mappings and receive regular feedback from advisors.	Information without correspondence or misleading conversion.

Probability: very low: lower than 10% low: greater than 10% and lower than 25% moderate: greater than 25% and lower than 50% high: greater than 50% and lower than 75% very high: greater than 75%	Effect: insignificant: does not affect project development tolerable: delays are within the contingency serious: will cause project delays catastrophic: project continuity threaten
---	--

2

Related Work

This Chapter reviews the literature, explaining in more detail the previous topics addressed in Chapter 1.1, e.g, services, cloud computing, SaaS, service description and service marketplace. Three main topics will be addressed in this Chapter. In Section 2.1, we will talk about service based-economy, of how ICT affected the way we see and use services. In the next Section 2.3 we analyse different service description languages such as WSDL, that rely on technical details, or USDL and Linked-USDL which rely on business, operational and technical details. In the last Section 2.4 we talk about the state of marketplaces, giving some examples, such as SAPO Services.

2.1 Service Based-Economy

Since the first industrial revolution in the XIX century, the manufacturing sector has been the main contributor to the world economy. Only in the middles XX century the service sector started to play a more considerable role changing this paradigm. It is important to note that the services have “long been perceived as non-innovative or technologically backward activities”. Only in the late of the century this kind of perception was beginning to change, thankfully to technology and innovation. Only recently was perceived the true impact of investment in this area. As an example, “on post-1995, the United States experienced another acceleration of productivity growth which Europe did not, and most academics and policy makers point to the sizable differences in investment in ICT between the United States and other economic regions as a key reason for this. In particular, much of it originated in industries that produced or used ICT intensively. Given the commonly accepted wisdom that investment in ICT generates economic growth governments initiated policies to foster the adoption of ICT”. Today, the service sector is one of the most, if not

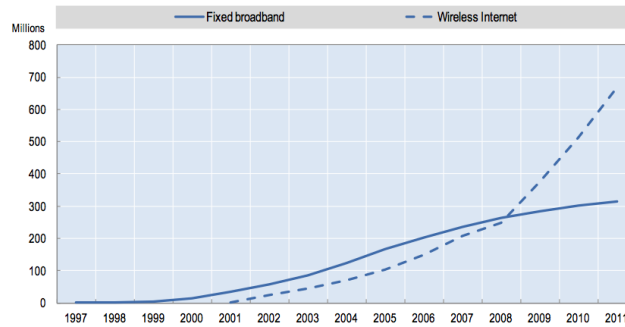


Figure 2.1: Wireless Internet access overtaking broadband subscriptions, in OECD Internet Economy Outlook 2012

the most, important aspects in the world economy, specially in the so called developed countries. They are the largest contributors of the value added for Economic Co-operation and Development (OECD) countries and through studies, we can see that this trend is to continue [44][24].

There are many reasons for the services growth, but one that we consider important is the impact that mobile devices and the Internet had. On today's most developed economies, most people can have several devices, like smartphones, tablets or computers, and a continuous internet connection, like mobile internet or using wireless networks. Compared to 2005, "the number of mobile phone subscriptions worldwide has more than doubled since 2005 and tripled in non-OECD countries". With so many people using devices connected to the internet, and the fast connections network improvements, the traffic on the internet has increased and is now growing at an average rate of 50% per year, accordingly to the last OECD report about this matter, in 2006. As a curiosity, "today, twenty households with average broadband usage generate as much traffic as the entire Internet carried in 1995" [54][20].

Trough the Figure 2.1, we can see that from 2008, wireless internet access grew exponentially, surpassing the fixed broadband. We can see, that today, the number of wireless internet access was more than double compared to the fixed broadband. This combined with the falling prices and better connections, is leading to huge level of Internet adherence.

2.1.1 How does the Internet influences the economy?

The Internet's communication enhancements are affecting economies at several levels and different areas. It brought huge benefits in consumer welfare. To enumerate some of these advantages, such as, a large variety of ser-

2.1. SERVICE BASED-ECONOMY

vices, improved communication, more distributed channels, lowered the prices, improved information gathering, a more efficient labour market, positives impacts in healthcare and education, and so on. The Internet brings some of these advantages to the Government, like a better and more transparent communication and information switch between their citizens, a fast provision of citizen rights through services, and much more. It is interesting to see that the Internet started with the objective to improve communication, and now it has been transformed into an ubiquitous technology [49]. If we go back, the first Internet adopters were the businesses, and they helped largely the increasing of Internet high speeds. Accordingly to the article [3], “in 2003, less than four companies out of ten companies had broadband access in the EU15; by 2009, this proportion had increased to nine firms out of ten”. As we can see in Figure 2.2, in 2011, almost every business had broadband access. The average being 88%, with Portugal getting slightly below this. Most of the companies had restructured their business models in association with Internet. Processes can be organized which allows for a reduction in capital needs through better utilization of equipment and resources. The increased communication can reduce costs and the number of supervisors [31]. This change led to improved efficiency and rapid growth of new online businesses. With the enormous amount of mobile devices, the business paradigm has changed, with the social networks or the video and audio services. In fact, most of businesses were forced to rethink their business models in order to survive to this new reality.

Thanks to the huge potential of Internet, there are infinity ways of providing a product or service. Right now, are emerging new products, typically not associated with communication capabilities. As example, refrigerators with Internet services, lights, televisions, and so on. This new paradigm is called ubiquitous systems, where information processing has been thoroughly integrated into everyday objects and activities. In the few years to come, its expected that every family could have one hundred devices connected to Internet, and with each other [41]. Its not difficult to see this scenario. Today, our smart-phone can control remotely a computer or a television, but if we consider that it can also control the lights, the temperature, the appliances, or, in the future the car, we started to gain consciousness that almost device will be connected.

This means that this sector is always changing and growing due to the technologies. Its fundamental that the services companies and the policy makers continue to invest, promoting ICT skills and employment measures in order to take benefit from the economic and social of these technologies.

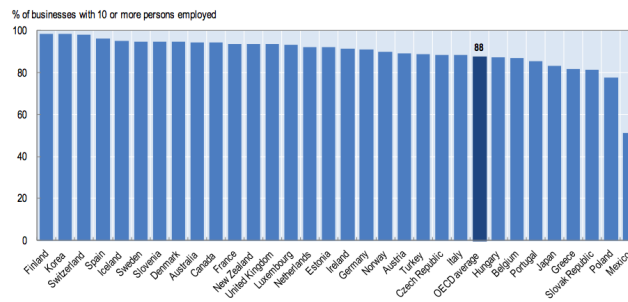


Figure 2.2: Business with a broadband connection in 2011, in OECD Internet Economy Outlook 2012

2.1.2 ICT and Services Employment Impact in Economy

After the discussed growth of ICT and the service sector, it is important to know the impact of ICT employment in economy. There has been a distinguished growth in ICT employment, which includes “employment directly in the ICT sector but also indirectly in terms of ICT-specialists in non-ICT sectors and also among ICT-intensive users in all sectors which rely on ICT skills to perform their work”. Because of the financial crisis, the ICT labour market has now been suffering like all other markets. However, the “recovery in ICT services employment and ICT-skilled employment has been much faster”. In contrast, ICT manufacturing employment continuous to decreased, mostly because OECD countries are outsourcing ICT manufacturing to non-OECD economies. Although it seems a very negative aspect in the economy, in practice, most countries can outweigh the employment loses in ICT manufacturing with the ICT services employment.

In Figure 2.3, we can see that the ICT sector contributes to a significant share of total employment. The average of OECD is nearly 6% where Portugal is one of the countries with lower share of ICT employment. If we compare to United States, the differences are even bigger. They have 30% of ICT employment in 2009, followed by Japan with nearly half the share. A country with few resources as Portugal, should seriously consider changing policies in order to increased ICT employment sector.

As showed in Figure 2.4, ICT services sector was the largest contributor to ICT sector in opposite to the ICT manufacturing. We can see that employment in IT services has grown more rapidly than the employment in business services.

As seen, services sector is growing at a considerable rate, leading to an increasingly tied service economy.

2.1.3 Services

In the section before, we explain the impact that services have in economies. But what is a service? Since the beginning of civilizations, services quickly become part of everyday life. They are so ingrained in our society that we often do not realize that we are using them. As example, we can rent a car, going to the hairdresser, watch a movie, teaching, these are all services. A service is a non-material equivalent of a good. It is also important to explain, that services can affect not only persons, but also goods. This could bring many advantages, as we mention before, to the peoples life [28]. With the exponential growth of ICT, the service concept has changed to a more technological side. This brought new services definition. Accordingly to W3C, “A service is an abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of providers entities and requesters entities. To be used, a service must be realized by a concrete provider agent”. This view is collaborated with other authors [48][34]. Today, we hear constantly the terms Service, e-Service and Web Service to refer the same thing and sometimes different things. It is important differentiate this concepts, in order to “conceptually separate and address the various stakeholders involved when architect an enterprise wide solution based on services”. According to this author, we can established a mapping between this concepts. Service is in the domain of Business Science, e-Service in Information Science and Web Services in the Computer Science.

Business Services. As we previous said in Section 1.1, a service is the non-material equivalent of a good. Business services are business activities provided by a service provider in order to create value to a consumer. This services are usually invoked manually but their realization can be manual or automated. As an example, hair cutting, teaching, plumbing repair, etc. We differentiate services from products due to their intangible form. Most products can be described physically based on their observable properties. Services in the other end, need to be described based on their effects in the consumer.

e-Services. When the Internet started to grow and evolve exponentially, most of the companies began to use “electronic information technologies for supplying services that were to some extend processed with the mean of automated applications”. In order to differentiate from other services types, the concept of e-Service appear to describe transactions conducted over the Internet. They are services for which the Internet or any other network are used as a channel to interact with consumers. It is important to mention that

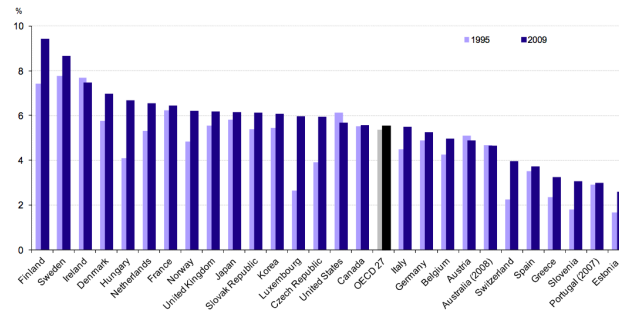


Figure 2.3: Share of ICT employment in business sector employment, 1995 and 2009, in [41]

those services are “independent of the specification language used to define its functionality, non-functional properties or interface”.

Web Services. The rise of technologies such as Service Oriented architecture [22] (SOA) and Service Oriented Computing [8] (SOC) caused services to be in the center of the technological revolution. With fast growth of web technologies the service concept evolved into the technological area, to the so called Web Services [10]. Accordingly to W3C¹ “A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with a XML serialization in conjunction with other Web-related standards”. Any e-Service that are available for consumers through Web-based protocols or web-based programs is a Web Service.

With this increase in technology and services a new paradigm has emerge. Cloud computing was the natural evolution and integration of advances in several fields such as distributed computing, service oriented architectures and utility computing. Cloud computing offered resources in an “economical, scalable and flexible manner”, which brought affordable and attractive advantages for businesses and consumers. Because of this, companies began to invest in new ways of reaching the consumer and one of those ways was through services in the cloud, as we state in Section 2.1.4.

¹<http://www.w3.org/TR/ws-gloss/>

2.1. SERVICE BASED-ECONOMY

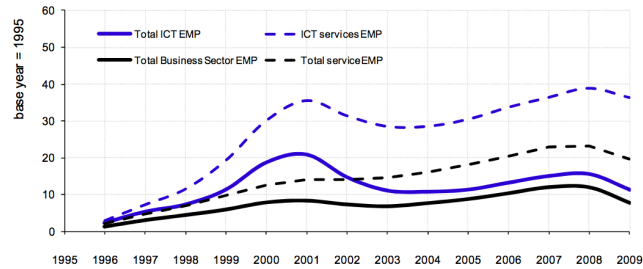


Figure 2.4: ICT sector employment in the OECD area by sector, 1995-2009, in [41]

2.1.4 Cloud Services

The emergence of Cloud computing is a derivation of the huge technological advancement in ICT. To move forward with this theme, we need to understand more accurately what is Cloud computing. Accordingly to NIST² “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” [37].

Nowadays, all the major companies, such as PT³, Amazon⁴, Microsoft⁵, and even the small and medium enterprises (SMEs) are moving to the Cloud. With this paradigm switch it is now possible to think in computing, services and storage as an utility that is provided on demand to anyone who wants to use it. This possibility are changing the IT industry, “making software even more attractive as a service and shaping the way IT hardware is designed and purchased” [5]. As an example, Developers no longer required large capital funds in hardware or software licences to deploy and operate their service. This also means that they can be focus on one job only, having much more time for improving their software. Companies with large tasks can get better results (if the software can scale) since using one hundred serves for one hour can cost almost the same as using one server for one hundred hours. Cloud computing have five essential characteristics accordingly to NIST:

- ***On-demand self-service.*** A consumer is able to request automatically computing resources as needed, without any human service’s provider interaction. The on-demand model was created to overcome the common

²url<http://www.nist.gov/index.html>

³<http://www.telecom.pt/InternetResource/PTSite/PT>

⁴<http://www.amazon.com/>

⁵<http://www.microsoft.com/pt-pt/default.aspx>

CHAPTER 2. RELATED WORK

challenge to an enterprise of being able to meet fluctuating demands efficiently;

- **Broad network access** Services capabilities are available through the network and are accessed by standard mechanisms which allowed the communication between different platforms, such as mobile phones, tablets or computers;
- **Resource pooling.** Because an enterprise's demand on computing resources can vary drastically from one time to another, it is necessary that the providers computer resources are available to serve multiple consumers with different physical and virtual resources in a dynamically way accordingly to the consumers needs, e.g, processing, memory, storage, virtual machines, etc. ;
- **Rapid elasticity.** Capabilities such as processing, storage, memory and so on can be rapidly and elastically provisioned. This allowed that this capabilities may be quickly scale out, and if necessary, rapidly released in order to quickly scale in. For the consumer, this capabilities are seen as unlimited, which may be requested at any time and amount;
- **Measured Service.** "Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service" such as processing, storage, memory, etc. These resources can be monitored and controlled in order to provide a better service to the consumer. This mechanism is completely transparent to the end user and provider.

It is also important to know that Cloud computing refers both to the applications delivered as services for the consumer as also the hardware and software in the datacenters, that are the support for those services. There are different categories of Cloud services such as Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS), Communication as a Service (CaaS), Monitoring as a Service (MaaS) [32]. We can have also anything as a Service (XaaS) and is a reference to either one or a combination of one or more of the previous categories of Cloud services described [42]. "XaaS is quickly emerging as a term that is being readily recognized as services that were previously separated on either private or public Clouds are becoming transparent and integrated".

In the context of our work, it is important to describe in detail what is a Infrastructure as a Service, Platform as a Service and Software as a Service. The others categories are out of scope.

2.1. SERVICE BASED-ECONOMY

IaaS. Infrastructure as a Service is when the consumer does not deal with the infrastructure, instead the responsibility of the equipment falls only on the Service Provider. He not only owns the equipment but will also be responsible for its running and maintenance. Accordingly to NIST definition, “The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls)”.

PaaS. Platform as a Service provides the capability for consumers to have applications deployed without the cost of buying and managing the hardware and software. Accordingly to NIST definition, ‘The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations”.

SaaS. Software as a service is the ability provided to the consumer to use on demand software that is supplied by the service provider on a cloud infrastructure via a thin client such as a web browser over the internet. The consumer does not manage or control the infrastructure (network, servers, operating systems, storage) or the application capabilities. Accordingly to [17] “is a multilayer model, existing of an infrastructure as a service (IaaS) and platform as a service (PaaS), complemented by the applications developed and owned by the service provider. The former component exists of networks, servers, storage and application programming interface, which are responsible for enabling convenient, on-demand access to computing resources. Enriching IaaS with middleware, that connects the application code with run-time infrastructure, one creates the PaaS. The latter allows interoperability between the infrastructure, various operating systems and hosted applications”. This can be seen in Figure 2.5 in order to better understand what is SaaS. Nowadays we are seeing a range of new functionalities, e.g, the possibility to write documents, edit photos or storage files without using any application installed in our computer. It can be quick and efficient delivery model for key business applications such as customer relationship management (CRM) or enterprise

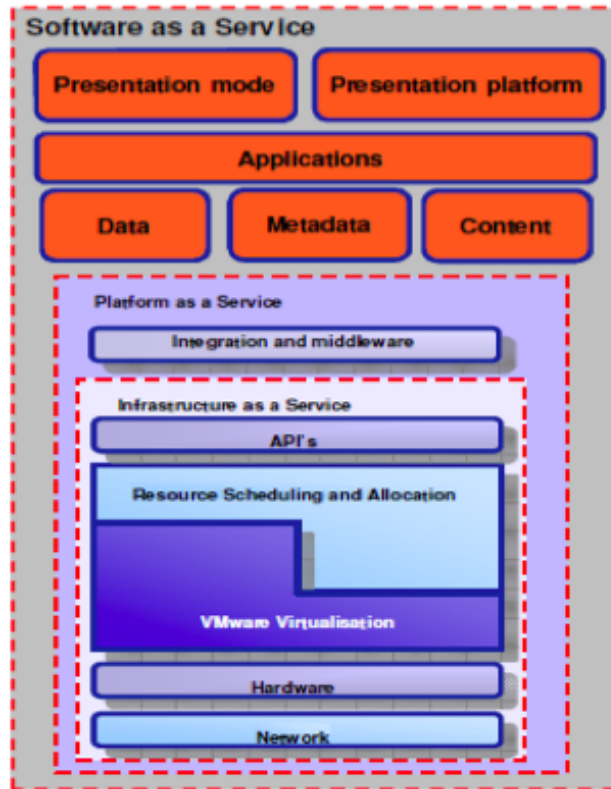


Figure 2.5: SaaS Multilayer Model [17]

resource planning (ERP).

The term Cloud is finally becoming into the minds of the people. We will see much more mature services offered in the Cloud, as soon as consumers fully understand their benefits.

2.3.

2.2 Service-Oriented Architecture (SOA)

Software developers have always struggled with the problem of creating software that is easy to maintain, extend and integrate with other systems. Many approaches in designing and writing software have been followed, such as modular programming and object orientation programming, until SOA approach appeared. SOA is a software design and an architectural style that supports service-orientation. This is a way of thinking in terms of services and service-based development and the outcomes of services. It defines how two computing entities, such as programs, interact in such a way as to enable one

2.2. SERVICE-ORIENTED ARCHITECTURE (SOA)

entity to perform a unit of work on behalf of another entity. One service can implement one action, such as make an order from an online shop were another service can verify if there are any available item in stock. These services use protocols to define how they send and parse messages using description metadata. The metadata should be understood and managed by the system designers, and also “software systems can use to configure dynamically by discovery and incorporation of defined services“ and to maintain coherence and integrity. As mention before, Web services can implement a service-oriented architecture. They can be one of the following: Service provider, Service broker or Service requester. The first one creates a Web service and publishes its interface and access information to the service registry. The second one “is responsible for making the Web service interface and implementation access information available to any potential service requestor“ while the last one locates entries in the broker registry using different search operations and then binds to the service provider to invoke one of its Web services. We can resume some SOA properties as mentioned below:

Properties

- ***Web services are independent.*** There are few requirements both on client and server side. On the client side, no additional software is required, only XML and HTTP are needed to start. On the server side it is required a Web service and a SOAP ;
- ***Web services can wrap existing applications.*** An existing application can integrate into the SOA with a simple Web service as an interface ;
- ***Web services hide implementation details.*** The only thing that the Service consumer needs to know are the interfaces to Web services. They are not concerned with the implementation details ;
- ***Web services are composable.*** We can aggregate simple Web services to more complex ones in order to perform high level business functions ;
- ***Web services are language independent and interoperable.*** “Client and server can be implemented in different environments. Existing code does not have to change in order to be Web services enabled.“ They mostly depend on HTTP and XML. ;
- ***Web services can be published, located, and invoked across the internet and are self describing.*** “Neither the client nor the

server knows or cares about anything besides the format and content of the request and response messages“. In order to perform this, some technologies are used such as HTTP, SOAP, WSDL and UDDI. ;

Nowadays, there are plenty of services available on the Internet to be consumed. But how can we capture all the relevant information about the service? In order to invoke and use services, it is necessary to capture this information with a service description language, as we will see in the next Section.

2.3 Service Descriptions

In this section we will show some of the existences description languages and ontologies for service description. Some of them can be define in tree perspectives, e.g, business, operational and technical, according to some authors. However, the technical perspective is the one who has known more developments. We will start for describing WSDL, USDL and then it will be presented the semantic approaches, like SAWSDL, WSMO, OWL-S and Linked-USDL. It is also presented, in Figure 2.10 the chronology of the different languages shown in this Chapter.

2.3.1 Web Services Description Language (WSDL)

WSDL is a model and a XML format developed by IBM and Microsoft for describing Web services “as collections of communication end points that can exchange certain messages” [21] and belongs to the group of SOA. It was developed with the purposed to describe the technical details for accessing a Web service providing the necessary information for invoke it from the developers point of view. However, the Ios as different requirements much broader than the ones described by WSDL. It needs operational and business perspective for describing services, e.g, it needs information about participants, price or other non-functional requirements.

2.3.2 Unified Service Description Language (USDL)

USDL is a data model for describing all kinds of services in every sectors of the service industry. The USDL family is composed by α -USDL, which is the original USDL version that was released in 2009 and can be seen in Figure 2.6, the USDL itself, which is on version 3.0 M5 and can be seen in Figure 2.7 and Linked-USDL which is based on Semantic Web Technologies [33].

USDL model tries to meet its requirements to unify modelling of services according to three perspectives, such as Business, Operational and Technical.

2.3. SERVICE DESCRIPTIONS

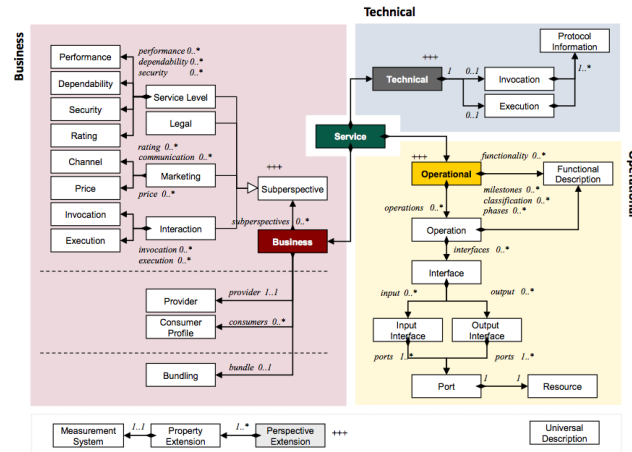


Figure 2.6: The α -USDL Model [12]

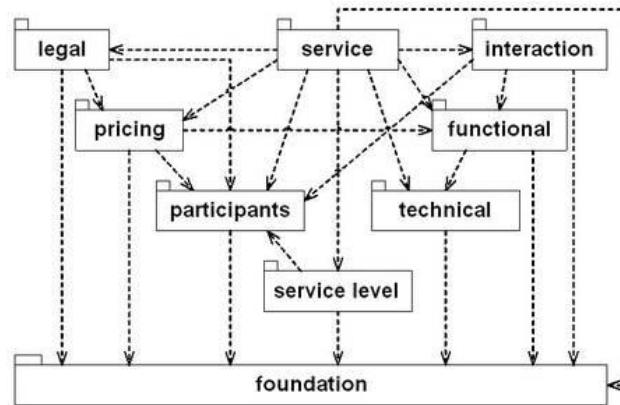


Figure 2.7: USDL Version 3.0 M5

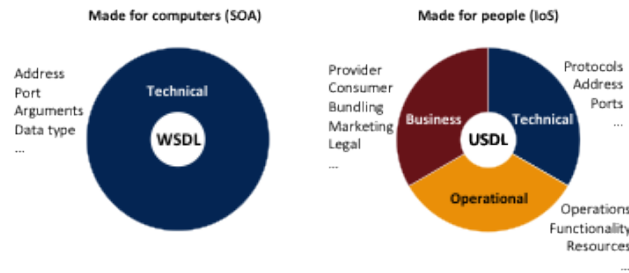


Figure 2.8: WSDL versus USDL [12]

The Business perspective describes the fundamental properties for characterizing a service, as such, legal, marketing, bundling, etc. The Operational perspective describes the operations that are executed by a service and the Technical perspective specifies how a service can be invoked and relies on reference to WS-* protocols. This service description changed the concept used in Web services and SOA service descriptions which only focus on development and technical aspects.

2.3.3 Semantic Approaches

When we use the Internet we are capable of doing some tasks such as search for a word meaning or searching for a shop that has good feedback and can delivery good prices. However, machines cannot accomplish all of these tasks without some kind of human direction. This happens because web pages are designed for people reading and understand it, not for machines to use it. The main purpose of the semantic web is enabling users to find, share and combine information. Their objective is to allow mechanisms so that the machines can readily interpreted information so that tasks such as finding and combine information can be done by them. This means that machines can "understand" and respond to the human requests based on their meaning. Next we presented some of the semantic approaches available, such as SAWSDL, WSMO, Linked-USDL, and so on.

Semantic Annotations for WSDL (SAWSDL)

SAWSDL [45], defines mechanisms using which semantic annotations can be added to WSDL components. It does not specify a language for representing the semantic models, e.g, ontologies. This approach tries to address the lack of reasoning capacity using logical inference. Accordingly to W3C, "these semantics when expressed in formal languages can help disambiguate

the description of Web Services during automatic discovery and composition of the Web services”.

Web Service Modeling Ontology (WSMO)

WSMO⁶ is a conceptual model that provide a Web Service Modelling Language (WSML) for describing semantic web services. This ontology evolved from the Web Service Modeling Framework (WSMF) [23]. The structure of WSMO is divided in four main elements: ontologies, goals, Web services and mediators [43]. Ontologies provide formal and explicit specification of the vocabularies used by the other modelling elements enabling automated processing of WSMO descriptions providing background knowledge for goal and Web services description. Web services are computational entities able to provide functionality and access to services. Goals describe “aspects related to user desires with respect to the requested functionality”. Mediators “decribe elements that handle interoperability problems between different WSMO elements. We envision mediators as the core concept to resolve incompatibilities on the data, process and protocol level”.

This language as any other has some problems. It lacks connection to other Web service standards. In order to outline this problem, was created WSMO-lite [52], a lightweight set of semantic service descriptions that can be used for annotations of WSDL elements using the SAWSDL annotation mechanism. Another disadvantage is that is still outside of the target of IoS.

Semantic Markup for Web Services (OWL-S)

OWL-S [35] is an ontology built on top of the Ontology Web Language (OWL) [36] which imports WSDL concepts. Is composed by three different parts as we can see in Figure 2.9:

- **Service profile.** Is for describing what the service does, and has the objective of advertising and discovering services. It has the non-functional requirements;
- **Process model.** It gives a detailed description of a service’s operation and to orchestrate him;
- **Grounding.** Provides details on how to interoperate with services through messages,

As we have seen in other service description languages, OWL-S provides the necessary structure to automate services discovery and automation but also

⁶<http://www.w3.org/Submission/WSMO/>

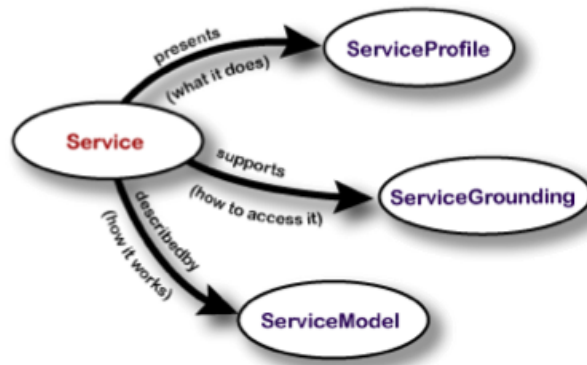


Figure 2.9: OWL-S service description elements [35]

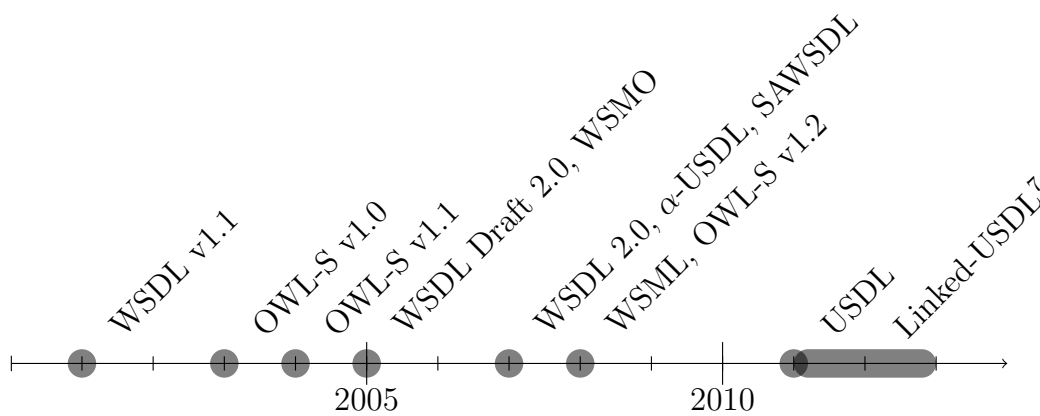


Figure 2.10: Services Description Language Chronology

to integrate them with the Semantic Web. It lacks a broader scope, as it only focuses on web services, however is a well founded base capable of attaining the three perspectives of IoS.

Linked-USDL

Linked-USDL is a new variant of UDSL, which tries to solve some previous issues on that model. The first problem found in UDSL, is its complexity. His view on services resulted in more than a hundred classes, taking the data model to an unwanted complexity level. Despite this complexity to describe a service, it was insufficient to describe all. New elements necessary could only be implemented by the UDSL Incubator Group. Another problem encountered was the amount of few real users that used this model which mean that it was

not adopted as desired, despite its potential.

Linked-USDL is a review of the previous model. It uses Semantic Web Technologies as its foundation, providing customizability and extensibility. The complexity of the model description decreased due to some aspects that are already available via Linked Open Data. Linked Data is “a term used to describe a recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF”. Using the Semantic Web Technologies, everyone can implement new elements, providing the means to describe specific domains. This already shows some potential over the previous model. The overall idea is simple, Linked-USDL tries “to develop an ontology to represent services by establishing explicit ontological links to other existing ontologies emerging from Linked Data initiatives”.

As is still in development not all the necessary modules are available. The ones available are: USDL-Core, USDL-Pricing, USDL-SLA and USDL-security. However, there is information that has been withdrawn, modified or does not yet exist on Linked-USDL. This is due to the fact that Linked-USDL is relatively recent, and as such, all modules are being revised. Some are yet to be created, such as: a vocabulary for some of the elements in Service Level. The only modules that are final are USDL-Core and USDL-SEC, so there is still much work ahead.

After we presented some languages to describe services the next step is to make them available in an easy way to the consumer. As we already saw, if services are not available in a specific mechanism such as a marketplace, the consumer has to spend a lot of time searching and comparing their characteristics. The services marketplace allow a consumer to invoke, search and compare services in an easy and fast way, as we will see next in Section 2.4.

2.4 Service Marketplaces

As previously mentioned in Section 1.1, once services are described, then they are suitable to be in one place to trade services. As we already know the mechanism that works in this direction is the marketplace. According to the author in [29], marketplaces are “an electronic platform for the exchange of information, products, services and payment via the Internet between buyers and sellers”.

Since marketplaces can bring together many buyers and sellers, and also provide the ability to automate transactions, it allowed it to reduce costs for both actors. But this is not the only benefit. According to the author in [29],

CHAPTER 2. RELATED WORK

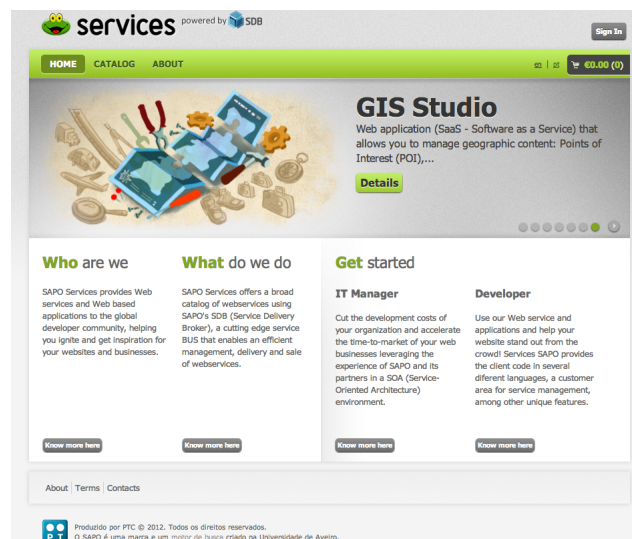


Figure 2.11: SAPO Services Marketplace

some of the benefits of marketplaces are:

- “Reducing search costs by facilitating comparison of price, products, and services;”
- “Reducing marketing costs compared to traditional marketing media;” due to the only distribution channel;
- “Operation 24/7 and round the clock on 365 days;”
- “Interacting more effectively in terms of services marketing communication;”
- “Enhancing customer relationships” due to the ease of communication.

As we can see, there are enormous advantages in having a marketplace, but they create value mainly through matching and aggregation. The matching mechanism is “required for spot sourcing where prices, contrary to systematic sourcing, are determined at the moment of purchase”. The aggregation mechanism implies that marketplaces aggregate consumers and providers under only one place, which “increases efficiency in procurement and reduces transactions costs”.

Is important to mention that there are different types of transactions in marketplaces, such as, Business-to-business (B2B), business-to-consumer (B2C), consumer-to-consumer (C2C), and is starting to appear business-to-business-to-consumer (B2B2C). B2B describes commerce transactions between

2.4. SERVICE MARKETPLACES



Figure 2.12: Amazon Web Services Marketplace [35]

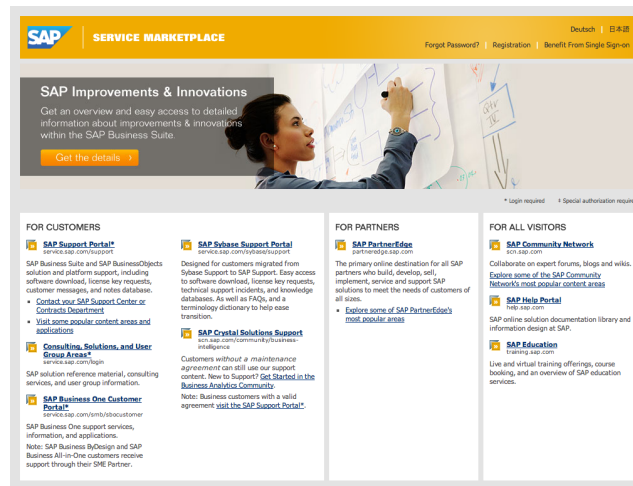


Figure 2.13: SAP Services Marketplace[35]

businesses such as between a manufacture and a middleman. The volume of B2B transactions are much higher than the volume in B2C. This has to do with the number of transactions performed, e.g, a cell phone company makes several B2B transactions such as buying chips, batteries, monitors for its phones. When a phone is sold to the consumer is only one B2C transaction. B2C describes commerce transactions between businesses and the consumer, e.g, a consumer buys something from a business entity. As an example, Figure 2.13 shows SAP services. It uses B2B transactions, since it only allows transactions between businesses. C2C “involves the electronically facilitated transactions between consumers through some third party”. Online auction, such as Ebay⁸ in which a consumer posts an item for sale and other consumers try to purchase it. The site only charges for a flat fee or commission. B2B2C groups B2B and B2C. With the same online platform and distribution channel, the objective is creating a complete value chain since the beginning of product or service creation until it reaches the consumer. As an example, Figure 2.11 shows SAPO Services marketplace. It uses B2B2C transactions, since it allows developers to get those API’s at a cost in order to develop applications for future users, or allows consumers to get those API’s for personal use. It has different services from different providers. The same happens to Amazon Services. Figure 2.12 shows Amazon Web services which lies on the same B2B2C transactions as SAPO Services. It provides services for companies as well as specific services for consumers. As an example, Amazon EC2⁹ can be used by consumer to buy instances of machines to process a certain job. The same way a business can do exactly the same.

According to the IoS vision, the marketplaces need to offer complex functionality to bring a large number of consumers and providers and in order to enable business interactions among them. Consumers initially can choose services from different providers based on functionality, pricing, community feedback, or other aspect of the service. After selecting the service, the consumer may have to pay an amount for the service consumption, so in the end the provider can delivery the service [14]. IoS, as we already mentioned, needs to have strong emphasis on the business and operation perspectives of a service. Accordingly to Cardoso [14], the following topics need to be analysed:

- ***Legal, Community Aspects and Business Models.*** “The implications of business value networks need to be studied from a legal perspective. The combination and integration of world-wide regulations is fundamental. A special emphasis has to be given to the generation of

⁸<http://www.ebay.co.uk/>

⁹<http://aws.amazon.com/pt/ec2/>

2.4. SERVICE MARKETPLACES

new business models for all stakeholders (i.e., service providers, aggregators, and consumers) and corresponding incentive mechanisms. Community aspects encourage cooperation and boost innovations through the extensive exchange of knowledge”;

- ***Service Search based on Advanced Service Description.*** “Marketplaces need to offer search mechanisms that allow for heterogeneous search criteria.” In order to do this, a language for describing different aspects of the services are needed. This description encompasses the functional and non-functional aspects, such as rating, security, pricing, etc. With this consumers will be able to search for services based on functional aspects as also on “natural based descriptions”. The non-functional aspects can largely improved the search results;
- ***Negotiation of Service Level Agreements.*** Before the consumer interact with the service, it needs to create an agreement with the service provider. This will allow to be addressed the rights and obligations that both parties have regarding the service;
- ***Service Monitoring.*** In order to maintain the quality, trust and so that providers can deliver the service under the conditions defined with the consumer, it needs to monitor the interaction;
- ***Billing and Payment.*** Consumers may have to pay for the services they used. This mechanisms, “may be provided either by the marketplace or by a third party”, need to be in place, so that the consumer may be charged correctly;
- ***Service Governance.*** “Governance addresses the strategic alignment between business services and business requirements thereby reducing risks and assuring compliance with rules and regulations. The ability to freely compose and orchestrate business functions which are available as services on a diversity of marketplaces bears overwhelming opportunities. Trust and trustworthiness of service offerings must be facilitated by the platform, balancing individual requirements, policies, and must be capable of adapting to the given business context”;
- ***Service Delivery Platform.*** In order to enable business transactions, an infrastructure for service delivery has to be provided.

2.4.1 SAPO Services Marketplace and their connection with SDB

SAPO services is a marketplace where user can buy Web services in a SOA environment and model them according to their needs, allowing a tremendous flexibility in the process of creating new applications. It has the purpose of providing Web services and content with great quality to the user such as providing “APIs and applications to programmers and other users, enabling the reuse of existing services in SAPO“, providing high quality, availability and performance Web services to the users and providing a large catalog of “multi-functional and multi-platform Web services that fulfil the needs of national and international“ developers. This brings many advantages to the software developers, e.g, allowing a lower development cost and also a reduction in the development time which enables faster delivering products. The large and diverse catalog also brings many opportunities to software developers and business owners.

Currently, SAPO services have thirty five services spread across different categories. These can range from social and communication to development and search. As mention previously, this could bring many flexibility to the developers. All of this services are supported through a Service Delivery Broker (SDB), which is a “middleware that enables the management, delivery and sale of webservices to certified users, in a SAPO controlled environment“ that allows service providers to expose services to third parties with entire control. This middleware has the following components as seen in Figure 2.14:

- ***SDB Runtime.*** This is the engine responsible for exchanging messages between service consumers and service enablers. It provides features to the service enablers such as transformation, routing, authentication, and others;
- ***SDB Support Application.*** This is a “web application that is built on top of the SDB Support Services enabling the management of the SDB runtime, MSS, ISS, the marketplace services, services SLA, product lifecycle and the services lifecycle. This application is crucial because it makes the enforcement of the patterns and best practices over the services and products“;
- ***SDB Marketplaces.*** This allows third parties to buy and sell their products. “On top of the marketplace services, was developed a web application that the service provider can use without having the need to develop his own marketplace“. It is important to note that he can also build and integrate the services on a specific context ;

2.4. SERVICE MARKETPLACES

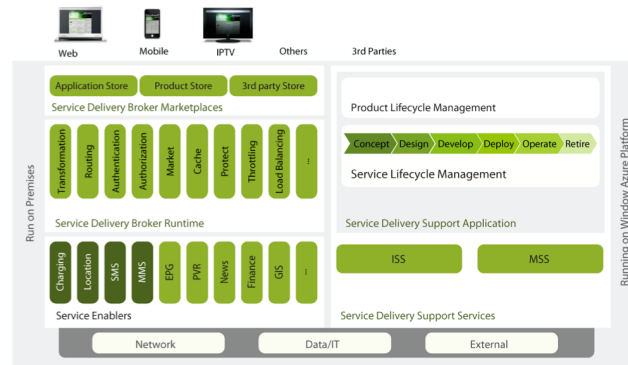


Figure 2.14: SDB Logical Architecture[4]

- **Service Enablers.** Is the set of services available such as GIS, MMS, SMS, and others, that expose their capabilities to the third parties. All of this services must “implement SES Management interface and the SDBA ensures that they are fully interoperable among all programming languages“.

3

Service Description Models

This Chapter gives an overview of the different models used in the project, e.g. SDB data scheme, α -USDL, USDL v3 M5 and Linked-USDL, such as their differences. Section 3.1 reviews SDB data scheme and gives an example of a service available at SAPO Services marketplace. In Section 3.2 we explain the α -USDL model so that the reader can understand the differences presented in the tables of the Section 3.4. Section 3.3 we talked about classes and properties available to describe services in Linked-USDL. The last Section 3.4 shows the differences between the two models, featuring some tables in order to better visualize it.

3.1 SDB data scheme

In this chapter we show the SDB data scheme and present some comments about this model. The scheme described is the last version received, on .

```
1 <Root>
2   <Service>
3     <Name></Name>
4     <Version></Version>
5     <Owner></owner>
6     <Status></Status>
7     <AccessLevel></AccessLevel>
8     <ResourceDescription></ResourceDescription>
9     <Parties></Parties>
10  </Service>
11  <Projects>
12    <Project>
13      <Name></Name>
14      <Version></Version>
15      <Owner></Owner>
16      <Status></Status>
17      <AccessLevel></AccessLevel>
18      <Parties></Parties>
```

CHAPTER 3. SERVICE DESCRIPTION MODELS

```
19 <Schemas>
20 <Name></Name>
21 <Version></Version>
22 <Owner></Owner>
23 <Status></Status>
24 <Parties></Parties>
25 </Schemas>
26 </Project>
27 </Projects>
28 <Operations>
29 <Operation>
30 <Name></Name>
31 <RequiresSSL></RequiresSSL>
32 <RequiresCredentials></RequiresCredentials>
33 <isOneWay></isOneWay>
34 <status></status>
35 <accessLevel></accessLevel>
36 <Parties></Parties>
37 <Strategies>
38 <clientId></clientId>
39 <Owner></Owner>
40 <Container></Container>
41 <Status></Status>
42 <File></File>
43 </Strategies>
44 </Operation>
45 </Operations>
46 </Root>
```

Listing 3.1: Simplification of SDB data scheme

As we see in Line 2, 11 and 28 in Listing 3.1, SDB data scheme have 3 primary tags for describing the service: *Service*, *Projects* and *Operations*.

The tag *Service* has information about the *Name* and *Version* of the Service, is *Owner*, the *Status* to see if the Service is active or not, the *AccessLevel* to see if it is Public or Private, a *ResourceDescription* of the Service and his *Parties*, who have access to the Service.

The tag *Project* within the *Projects* is an entity that represents the documentation of the Service, used to produce the WSDL document. As some tags have the same meaning, it is not necessary to explain them again. The different ones in *Project* are: the Project responsible, *Owner*, the *Parties* with their different roles in the Project, and the new tag *Schema*, who have the same information has before but relatively to the *Schema*.

The last one, refers to the *Operations* of a Service. This one has the tag *Name* that refers to the operation name. *RequiresSSL* indicates whether the operation need SSL or HTTPS, *RequiresCredentials* indicates if the operation need credentials while *isOneWay* indicates if the operation is one-way. The *Parties* has a different meaning, are the different roles people that have access to the documentation of the transaction and its strategy. *Strategies* are the strategies of the Service. Has a specific account with access to the strategy,

clientId, the responsible for that strategy, *Owner*, and a *XML File* with that strategy.

As seen, the data scheme is aimed at the development cycle of a service. This is not ideal since the Linked-USDL does not provide a way to model this information. To outline this problem will be conducted a study in order to propose an extension to the Linked-USDL so that may be possible to model this information. This extension will not be addressed at this time. In the next section we will presented some of the domains in α -USDL.

3.2 USDL Model

In Chapter 2.3.2 we tried to explain in general what was the α -USDL model. As such, at this point we are more interested in describing and analysing their different domains as shown in Figure 2.6. Accordingly to the IoS vision, services need to be described not only through technical details, but also through the business and operational as we already seen in α -USDL.

3.2.1 Business

The business perspective relies on a set of different non-functional properties such as security, pricing, bundling, obligations and rights, etc. This part is fundamental for describing services because encompasses information which is easily understood by the business stakeholders and consumers.

As it is quite complicated to have a language that describes all the different characteristics of services in different branches, the α -USDL includes an extension mechanism that allows creating new domains with new properties. Next we will be explained the different Business perspectives.

Service Level Domain. This Domain represents all the quantitative and qualitative characteristics of a service, which are required to meet all the requirements of consumers, and are also needed for the development and evaluation of SLA's. Comprises Performance, Dependability, Security and Rating.

1. Performance is defined as the time delay in a service invocation and execution phases. "Invocation response time describes the service's ability to respond to a service request within a specified time frame in the invocation phase. Execution response time states how long it will take for a service to fully complete service execution";

CHAPTER 3. SERVICE DESCRIPTION MODELS

2. Dependability is the ability to deliver a service that does exactly what is specified in its functional part. α -USDL enables service dependability using two different attributes: availability and reliability. The availability is defined as “the ratio of time period in which a service is available for execution”. The reliability of a service shows us in which degree a service is successfully fulfilled in a given time according to an agreed contract;
3. Security refers to the safety of the interaction channel used for the exchange of resources between the consumer and the service provider. The security area is very important for both actors. The consumer will only use the service if security is guaranteed and as such is of great importance that the provider should be aware. In USDL the goal of security is to ensure authentication, confidentiality, integrity and non-repudiation of resource exchange;
4. The Rating provides a feedback mechanism. This is one of the most important factors for a Marketplace to establish levels of trust between providers and consumers. There are two types of rating according to the USDL: Community and Expert. Community rating is the users opinions about a service, e.g, we can see this on sites like the Amazon¹ where users can vote in one to five stars and also leave some comments. Expert rating provides a objective view from an expert perspective, such as critics who made reviews on independent parties.

As we can see the Service Level have four different characteristics to evaluate the overall quality of a service. It is not suitable to judge the quality by analysing only one attribute. The combination of those four characteristics is what gives us the overall quality of the service.

Legal Domain. The USDL provides terms of use and clauses to formalize rights, obligations and penalties for the Consumer and Provider. The terms of access define the terms and conditions that a consumer must access to use the service. Clauses, are more related to the legal field, grouping these settings by functional aspects. Rights describe the legal or moral right to do or refrain from doing an action. An obligation refers to the behavior that is expected or required from the provider or consumer. Penalties are imposed on any party in the case of violating rights or obligations.

Marketing Domain. This part deals with the implementation of business strategies as part of advertising and sales. It is compound by three domains, such as pricing, documentation and certification.

¹www.amazon.com

Regarding the different pricing models, USDL supports:

1. A flat-rate , also known as a flat fee or linear rate, refers to a pricing structure that charges a single fixed fee for a service, for a given period, regardless of usage. It refers to a single fixed fee charged for services, irrespective of how the service is used. As an example, we pay the same amount of money for the Internet usage, no matter what days we use it;
2. In pay-per-use, consumers were charged for each unit they consume, as such the telecommunications were we are charged for a call;
3. A two-part tariff is a pricing model in which the price of a service is composed of two parts: a flat-rate fee and a pay-per-unit fee. In amusement parks consumers pay a one-time access in order to enter the installations, and a per-unit price for each service they consume on the park;
4. A variable tariff refers to a structure that indicates a minimum and maximum value for the price of a service. The actual price will be set by the provider after some kind of negotiation;
5. Finally we have the commission. This is a fee a broker charges for a transaction. This is used almost in every real estate when a agent sells a house gaining a commission for it.

Marketing also provides a documentation module. On this part is provided the Service documentation. This information can be provided by the service provider or the online community. E.g. It can be a user guide or technical documentation.

The last domain is the certification where is were the service provide information about their knowledge and experience in a particular area.

Participants Domain. The USDL identifies three types of stakeholders:

1. A service provider is an entity that offers services to consumers possibly in a marketplace. Each service provider attempts to promote its services to potential consumers in order to maximize its profit. The class includes organizational information of the provider and the identity of the responsible person for providing a specific service;
2. A service consumer may be enterprise systems, human users, mobile agents or any other systems that adopt the service and marketplace paradigm. They have all in common that they have a problem to solve and a goal to reach. Therefore, this class captures information that

CHAPTER 3. SERVICE DESCRIPTION MODELS

describes the profile of consumers that may be potential clients for a service. This class enables consumers to explore the use of problem-oriented and goal-oriented approaches to determine if a particular service has a potential that suit its needs;

3. A Partner is an entity that is involved with service provider in order to delivery services. Three types of contracts exist: partnership, alliance and other. A partnership is a formal association between two or more parties that have agreed to work jointly in the pursuit of common objectives. The agreement targets to joint several origination's funds, skills, resources and talents to provide a service and share the profits and losses. An alliance between businesses is a commitment, trust, and mutual agreement to benefit both parties. It is usually motivated with the objective to increase customer satisfaction and reduce costs. Other is used to refer to any other form of relationship not covered by partnerships and alliances.

Bundling Domain. A bundling is a set of services that are assembled because they provide an added value to the consumer as a whole. Two types of bundling are supported: pure bundling and mixed bundling. In pure bundling the consumer has no choice but to buy both services in the bundling. In the mixed bundling have the choice to buying bundled services or just buying one service without the others.

3.2.2 Technical

This USDL perspective has two modules. The Invocation part describes the protocols used for the Services invocation, while the Execution part describes the protocols used to implement services. These included a number of different attributes, such as transport protocols, messaging protocols, security protocol, reliable messaging protocols, transaction protocols, management protocols, user interface and metadata exchange protocols.

3.2.3 Operational

This perspective describes the elementary operations or activities offered by the services. It provides consumers with an understanding of what the service is providing from an operational perspective and, thus, what a consumer can expect from a service. It is compound by:

1. Functionality. These descriptions are a central pillar of services. It ignores details on how to invoke and execute a service and provide a black

- box description for determining the usability of a service;
2. **Classification.** This allows associating a service or its operations with one or more industrial categorizations based on standard taxonomies. Classifying services supports potential service consumers to discover services more efficiently and service consumers are enabled to find all suppliers of a specific type of service. Classification provides a mechanism that may be used by service providers and marketplaces to classify services with information that categorizes them according to taxonomies;
 3. **Phases.** Services need to be planned and organized with a predefined sequence of phases that needs to be followed in order to ensure that the services are to be executed successfully;
 4. **Milestones.** While a phase marks the beginning and acts as the container for operations and an operation is a specific action that needs to be executed within a phase, a milestone marks the end of a phase and all the operations within. Milestones provide a way to express the major states that a service will reach during its execution.
 5. **ConsumerProcesses.** The business process models can describe the orchestration and choreography of a specific service. The orchestration refers to the executable process model while the choreography identifies the message sequences between consumer and provider which are need to trade a service;
 6. **Operations.** This class represents the major operations that a service provides. Operations are also seen as black boxes. An operation allows a consumer to access services functionality. This allows service consumers to access particular subsets of services operations.

3.3 Linked-USDL Model

In the same way that we already described, α -USDL in Section 2.3, we will just explain in more detail the existing modules of Linked-USDL.

As is still in development not all the necessary modules are available. The ones that are available are: USDL-Core, USDL-Pricing, USDL-SLA and USDL-SEC.

USDL-Core. This vocabulary covers the core aspects of services. In Figure 3.1 we can see that this module could represent the description of the core attributes of services and their relationship. Also the involved parties in Service

CHAPTER 3. SERVICE DESCRIPTION MODELS

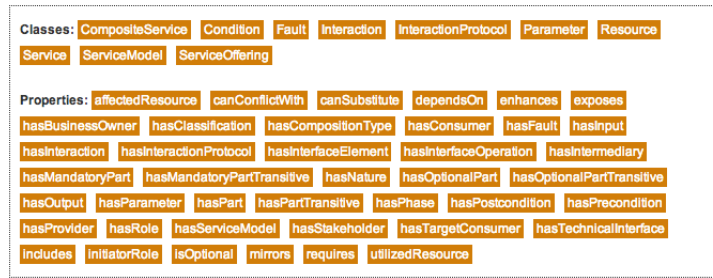


Figure 3.1: USDL-Core

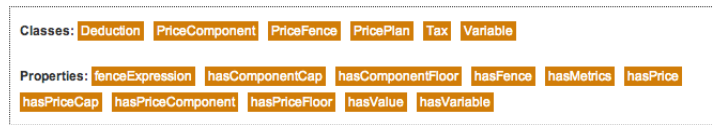


Figure 3.2: USDL-Pricing

delivery and how they interact with each other, and the description of this interactions. Specific aspects regarding some particular dimensions such as technical interfaces or security are leaved aside. These aspects will be covered in additional modules or by existing vocabularies.

USDL-Pricing. This vocabulary covers the pricing aspects of services. With this, it is possible to describe what is priced to witch amount and unit. Every Service has a *PricePlan* witch may consist in many *PriceComponent* as we see in Figure 3.2. As the complexity of describing the price of a particular service could be quite high, this module is still in development.

USDL-SLA. This vocabulary covers the Service Level aspects of services. As shown in Figure 3.3 there is two primary classes, one to *Guaranteed States* and other to *Guaranteed Actions*. The first one are Service Levels that must be guaranteed while the service is delivered. The second one are actions that are performed when a certain condition is fulfilled.

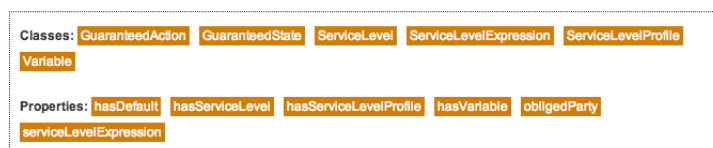


Figure 3.3: USDL-SLA

USDL-SEC. This module describes the main security properties and features of a service. With this, service providers can use this specification to describe the security features of their services. As already seen in Chapter 2.3 there are information that has been withdrawn, modified or does not yet exist on Linked-USDL. This is due to the fact that Linked-USDL is relatively recent, and as such, all modules are being revised. Some are yet to be created, such as Service Level. The only one that is final is USDL-Core, so there is still much work ahead.

3.4 Mismatch

Now that the three models were explained, a summary will be presented, along with a table that compares SDB Data Schema, α -USDL and Linked-USDL. As we can see in Table 3.1, a direct comparison is made between all the modules.

Table 3.2 gives an overview of the elements available in each domain in the different models. The interesting part of this is to see the differences between the number of elements in SAPO data scheme and the other service description languages, which are still large. The comparison between α -USDL, USDL v3.0 M5 and Linked-USDL only serves as a curiosity, since the fact that a model has more or fewer elements does not reflect in the quality of the model.

Service Level. The version received of SDB data scheme contains almost no relevant information. It has only information about the security. As we know it is very important to have information available of this module in order to properly implement SLA's. Other reason for that, is the possibility of having a system of ratings, which will allow consumers to give and receive feedback from a particular service, fostering the creation of a community.

Marketing. As we have already seen the marketing is divided into three modules:

1. Pricing. As we can see, there is no information regarding the Service pricing. However, based on the website it is possible to get this kind of information. This means that this could be provided in a different version of SDB data scheme;
2. Documentations. There is no information regarding this part. However, it is possible to obtain this information from the website. This means that this also could be provided in a new version of SDB data scheme;

CHAPTER 3. SERVICE DESCRIPTION MODELS

3. Certification. There is no information regarding this part and its not available on website.

Participants. This domain is divided in three modules:

1. Service provider. There is enough information concerning this part of service provider. The scheme makes reference about Parties, although, is essentially about the Service life cycle. This kind of information cannot be model in Linked-USDL;
2. Consumers. As we can see, there is no information regarding this module;
3. Partners. There is no information about Partners. As mentioned above, the Parties described in SDB data scheme essentially refer to people who intervened in the Service development cycle.

Legal. SDB data scheme doesn't have any information about legal aspects. However, on SAPO Services we could get some of this information.

Bundling. There is no information regarding the bundling part of the Service.

Technical. As services are described through the principles of SOA and using WSDL, it is normal that this domain has some information described.

Operational. This is the other module that can be populated via SDB data scheme, because it has some elements available.

Table 3.1: Differences between schemes

		SDB Data Schema	α -USDL	USDL v3.0 M5	Linked-USDL
Service Level	Performance	×	✓	✓	×
	Dependability	×	✓	✓	×
	Security	Partial	✓	✓	✓
	Rating	×	✓	✓	×
Marketing	Price	×	✓	✓	✓
	Documentation	✓	✓	✓	✓
	Certifications	×	✓	✓	×
Legal	Legal	×	✓	✓	✓
Participants	Provider	✓	✓	✓	✓
	Consumer	×	✓	✓	✓
	Partners	×	✓	✓	✓
Bundling	Bundling	×	✓	✓	✓
Technical	Invocation	✓	✓	✓	✓
	Execution	✓	✓	✓	✓
Operational	Operations	Partial	✓	✓	✓
	Functional Description	✓	✓	✓	✓

Table 3.2: Number of elements in each domain available in these service description languages

	SDB Data Schema	α -USDL	USDL v3.0 M5	Linked-USDL
Service Level	2	10	19	4
Marketing	1	9	19	6
Legal	0	4	14	6
Participants	1	3	8	6
Bundling	0	2	4	2
Technical	2	3	10	1
Operational	4	8	4	4

3.4.1 Mismatch Conclusion

Before going directly to conclusions is shown a brief summary of what type of data is available through data scheme and SAPO Services website:

SDB Data Schema

1. Brief description of the service (name, version, status, access level and description);
2. Parties. From what has been seen most are about development cycle, so they are not relevant to this case;
3. Information about security and operations (requireSSL, requireCredentials, access level, description, state).

Website Information

1. More specific Service description (features, advantages, and so on);
2. Price;
3. Some information about the website Coverage part, which may ultimately result in SLA's.

As already mentioned, few information is available in order to model a service. For IOS as well as for consumer is in their interest to have the Business and Operational perspective well defined. This will be the main approach to be used when a consumer is looking or want to compare a service. As such, it is necessary to have the most well described modules of this field. The Service Level Domain is essential to get services metrics. This allows the Provider to know Service performance, and therefore essential for the development and evaluation of SLA's. As we known, SLA's are critical to any business which it is through this that are defined the commitments between the service provider and the consumer. For participants, only the Service Provider is described in SDB data scheme. As such, it would be interesting to have a profile and Consumer Partners involved in this description. If this information were present, it would be treated in various different ways on Marketplace. As mentioned previously, the information provided by SAPO, is related to service development cycle. This type of methodology is beyond the concept of IOS, and as such, the Linked-USDL. However, as will be necessary to do Services mappings, it will be proposed an extension of Linked-USDL so that can be added the service development cycle. However, we must emphasize that it will be very interesting if i could have Business service information in SDB data scheme in order to describe more complex and complete services.

4

Analysis and Specification

This Chapter will describe the analysis and techniques that were used for the system definition. The first Section, 4.1, we will show the requirements, use cases and mockups of the interface. The Section 4.2 describe the system architecture.

4.1 Requirements Analysis

Requirements analysis is essential to the success of a system. It is necessary to have previously documented all the system characteristics, functionalities and limitations in order to facilitate system implementation. Requirements can be categorized in two major groups according to Sommerville[46], User requirements and System requirements. The first requirements are more oriented to what the user will be able to do with the system. The System requirements are more detailed descriptions of the system's functions and constrains. We can classify the requirements in two major groups:

- ***Functional Requirements.*** These requirements must explain how the system must behave or how he must react to a specific input, etc;
- ***Non-functional Requirements.*** These requirements are related to system constrains, such as time constrains, development process constrains, constrains imposed by standards and others.

4.1.1 Requirements List

All the requirements are listed in Appendix B were divided according to FURPS+[1] Methodology and prioritized using MoSCoW method[18] which has four levels of priority:

- *Must.* have the requirement;
- *Should.* have the requirement if possible;
- *Could.* have the requirement if there are enough resources and time;
- *Won't.* have the requirements in this system version or may have in the future developments.

We divided all the requirements according to the FURPS+ methodology which are: Functionality, Usability, Reliability, Performance and Supportability (FURPS) and a few more such as Design, Interface, Physical and Implementation (+).

We only use some of the categories from this methodology, because, there are no constraints of performance or also physical constraints.

4.1.2 Use Cases

In the Unified Model Language(UML), Use Case diagrams model the behaviour of a system and supports the capture of system's requirements. These diagrams identify the interactions between the system and its actors allowing them to describe what the system does and how the actors use it. They are very useful to model a business so that "all participants in the project share an understanding of the workers, customers, and activities of the business" and also to capture requirements, identifying classes and tests for the system.

Use Case A, Figure4.1, describes which actions the stakeholders may perform in the Web interface to perform the Export. They can load Files, Select the Export button to make the Export and then Download the resulting file.

Use Case B, Figure4.2, illustrate which actions the stakeholders may perform in the search page. They can do a search and then visualize the results.

4.1.3 Interface Mockups

A Mockup is a model used to demonstrate, evaluate and promote a concept. The most common use of Mockups is through user interfaces that show the user how the software will look without having to build the software. In this particular case we will have three different Mockups: Figure 4.3 represents the system login, Figure 4.4 describes the Export and the Figure 4.5 details the search. In the first Mockup we choose to have a login because the project are available in a public Web page. This means that only authenticated users can view the other project pages.

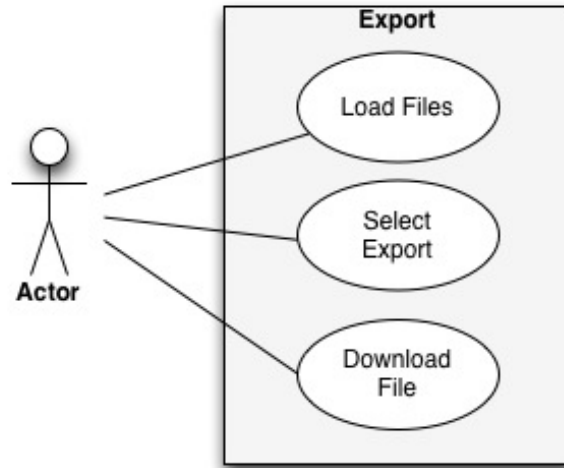


Figure 4.1: Use Case A - Export

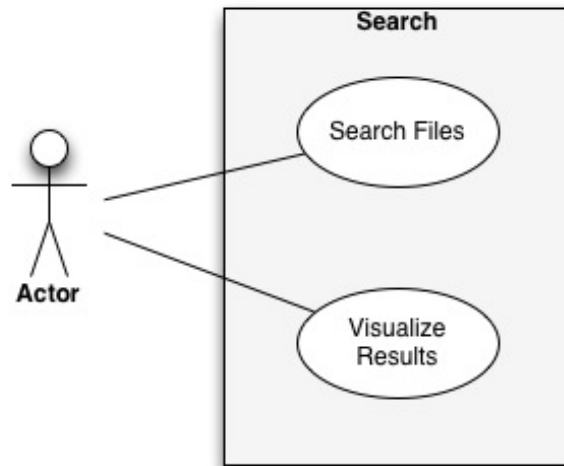


Figure 4.2: Use Case B - Search

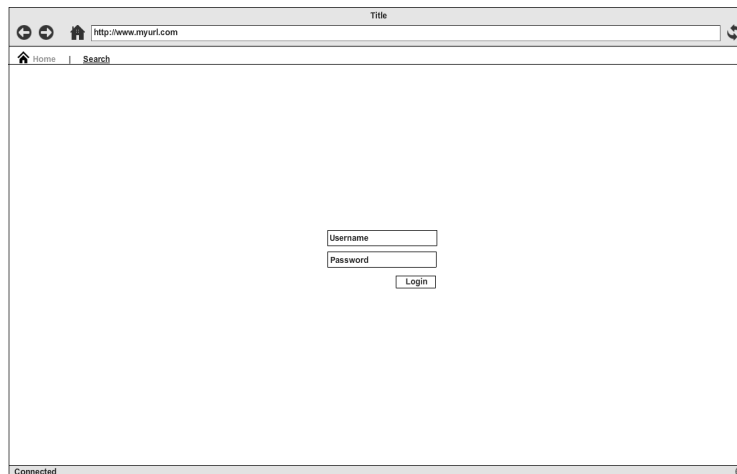


Figure 4.3: Mockup A - System Login

The second Mockup details all the necessary elements to perform the Export. We have two Textboxes where the left one will display the Export loaded file and the other will display the resulted file. There are three different buttons, one for load the file, the other for start the Export and the last one allows the user to download the resulting file.

The last Mockup represents not a project requirement but an extra for this project. It will allow the user to make three distinct searches. Two of which use a Google plugin to search SAPO services and search the Web, and the other will allow the user to search for a few Linked-USDL services enriched with a specific Taxonomy.

4.2 SUnSET Architecture

Before we started on the designing process it was necessary to choose firstly a modelling language that could be understood by all the interested parties. The first language that was considered was the Unified Modelling Language[9] (UML) mainly because is one of the most well spread and widely used tool in software projects. However, this language is too focused on software-based principles. We need something that could be easily described even for those who are not software oriented people.

From our perspective, the best specification to be adopted for describing our system's architecture was the Fundamental Modelling Concepts[30] (FMC). Their goal is to easily provide the concepts to create and visualize models so that all the stakeholders may have the same understanding of the system's

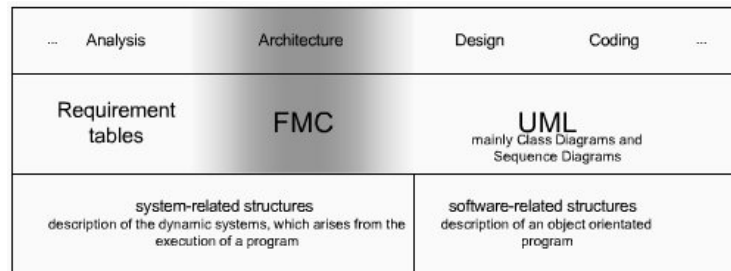


Figure 4.6: Software Life Cycle - How FMC fills the gap[30]

purpose. “FMC is based on strong theoretical foundations, it has successfully been applied to real-life systems in practice (at SAP, Siemens, Alcatel, etc.) and is also being taught in software engineering courses at the Hasso Platter Institute for Software Systems Engineering“[2].

These two tools (UML and FMC) may be considered as complementary such as both focusing on different phases in the software development lifecycle process as shown in Figure 4.6. Although UML provides a complete description of the software systems with diagrams such as Use cases, Class diagrams, sequence diagrams, etc, FMC fills the void left by the UML on describing system-related structures.

We choose than to use FMC for describing the system architecture. This specification have three different type of diagrams: Entity relationship, Block diagrams and Petri nets where we will use the Block diagram notation for our prototype. Accordingly to the reference sheet there are three different basic elements:

- **Active system component: agent, human agent.** “Serves as well-defined purpose and therefore has access to adjacent passive system components and only those may be connected to it“. They are represented by rectangular shapes;
- **Passive system component: storage, channel.** A storage is used by agents to store data and is represented by round shapes;
- **Access type.** “Directed and undirected edges represent the kind of access an active system component has to a passive system component.“ There are different types of access such as read, write, reply and request, and they are represented by lines and arrows.

With this basic elements it is possible to create any common and complex structures.

4.2. SUNSET ARCHITECTURE

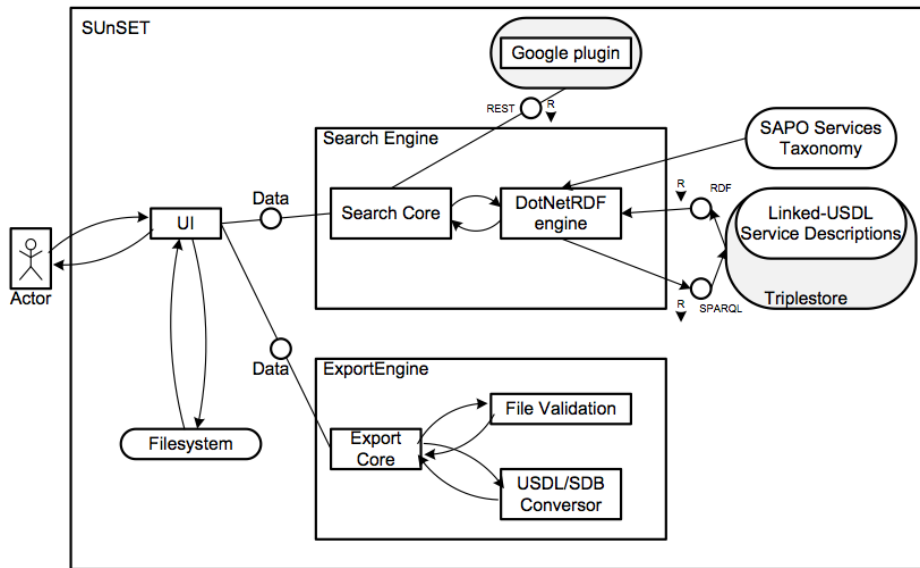


Figure 4.7: SUNSET architecture

In the next Sections we will explain the architecture design and the decisions that led to the final design.

4.2.1 Export Engine

Before we start to explain the Export Engine it is important to emphasize some particular aspects. First, as seen in Figure 4.7, there are two main components: Export Engine, which is the main purpose of this project, and the Search Engine, which is an extra to explain how Linked-USDL may work with a Taxonomy to describe services. There is also a peculiarity regarding the user interface and the database. This was only implemented so that only authenticated users could access the Web site.

The Export Engine component is intended to receive data from the User Interface, process this data and returning the resulting information again to the User Interface. The first step is analysing the data received and verify if this is valid through a combination of XSDs. If there are some invalid files the Export Engine will skip them and make the conversion only for the valid files. When this process is finished the Export Engine will forward the resulting file to the User Interface.

4.2.2 Search Engine

As previously stated this is an extra added outside the requirements. The Search engine component has a simple goal: search for a few SAPO services described in Linked-USDL and enriched with a specific Taxonomy. This component receives the data placed by the user and through the DotNetRDF engine it will search for services in the Triplestore. If they exist, the engine will return a list of results and send them to the User Interface.

As seen in Figure 4.7 we also used an API from Google in order to do a search in SAPO's services Web site.

4.2.3 Web Interface

Is through the Web Interface that the stakeholder interacts with the system. As mentioned previously this allows the stakeholder to perform the Export and a Search.

The Export User Interface has two textboxes where the stakeholder may or may not change the text before he decides to make the Export or download the resulting file. It has an option to upload files which then it places the main one in the first Textbox and the resulting file on the second Textbox. When the Export is completed the stakeholder has an option to download the resulting file to their remote computer.

The Search User Interface has only one simple search box and three checkboxes which allows the stakeholder to choose one of the three different options: search SAPO services, search the Web and to do a semantic search. The first two options use the Google plugin do to a search on Web where the last one uses the DotNetRDF library and a Taxonomy to search for services in the Triplestore. When there are something to return the data is presented sequentially to the stakeholder.

5

Semantic Models

In this Chapter are presented two semantic models. Section 5.1 presents and discusses an extension possibility for Linked-USDL which is the Service Lifecycle Ontology. Section 5.2 presents a Taxonomy of concepts for describing SAPO services.

5.1 Service Lifecycle Ontology

The Service Lifecycle Ontology aims to fully describe the lifecycle of all Services (SaaS) available today. It provides a set of Classes and Properties which will allow to represent Phases, Tasks, Roles and other information relevant for the Service Lifecycle.

5.1.1 Motivation

Through this thesis we found that there is a need and a concern for an organization to management their services lifecycle. Although we can use Linked-USDL to describe this services there is no mechanism to support that kind of description.

In order to fill this gap we decided to create an extension which will be described in the next Section.

5.1.2 Challenges

One of our major concerns was to create an extension that could encompassing all services lifecycle of each organization. It was also a challenging to know what kind of information was needed to describe a service lifecycle due to some different requirements through the readings.

In the end we decided to create an extension that was easy to understand but at the same time the most generic and complete as possible.

5.1.3 Model

Before we started to describe the model is necessary to have a comprehensive view of how service lifecycle works. In Figure 5.1 we present an example/overview of a service lifecycle. As seen, the service may have different phases, such as service deployment, service discovery, etc, and which there must be a transition to change his phase. Inside his phase and in order to have a transition, there are a group of tasks that need to be performed and which are decomposed into tasks to be conducted by one or more people.

With this brief summary it would be easier to understand the classes and properties of this Ontology.

5.1.3.1 Model Specification

This model has eight classes as seen in Table 5.1 and eighteen properties as represented in Table 5.2.

As previously mentioned this Ontology allows to describe an Activity which is when a person is associated and starts a task, a Phase where the Service is at the moment, a Role or a set of Roles involved in a Phase or other component of the Service, a ServiceLifeCycle where we can define which phases existed on that specific service, a Task and a TaskGroup where we can define the things we need to do on that phase, a Time for each Task or Activity and finally a Transition for describing what is the next phase and what is needed to succeed that.

On Figure 5.2 we can see the relationships between classes and properties and the appendix C gives us a complete description in Turtle of this Ontology.

Table 5.1: Service Lifecycle Ontology Classes

5.1. SERVICE LIFECYCLE ONTOLOGY

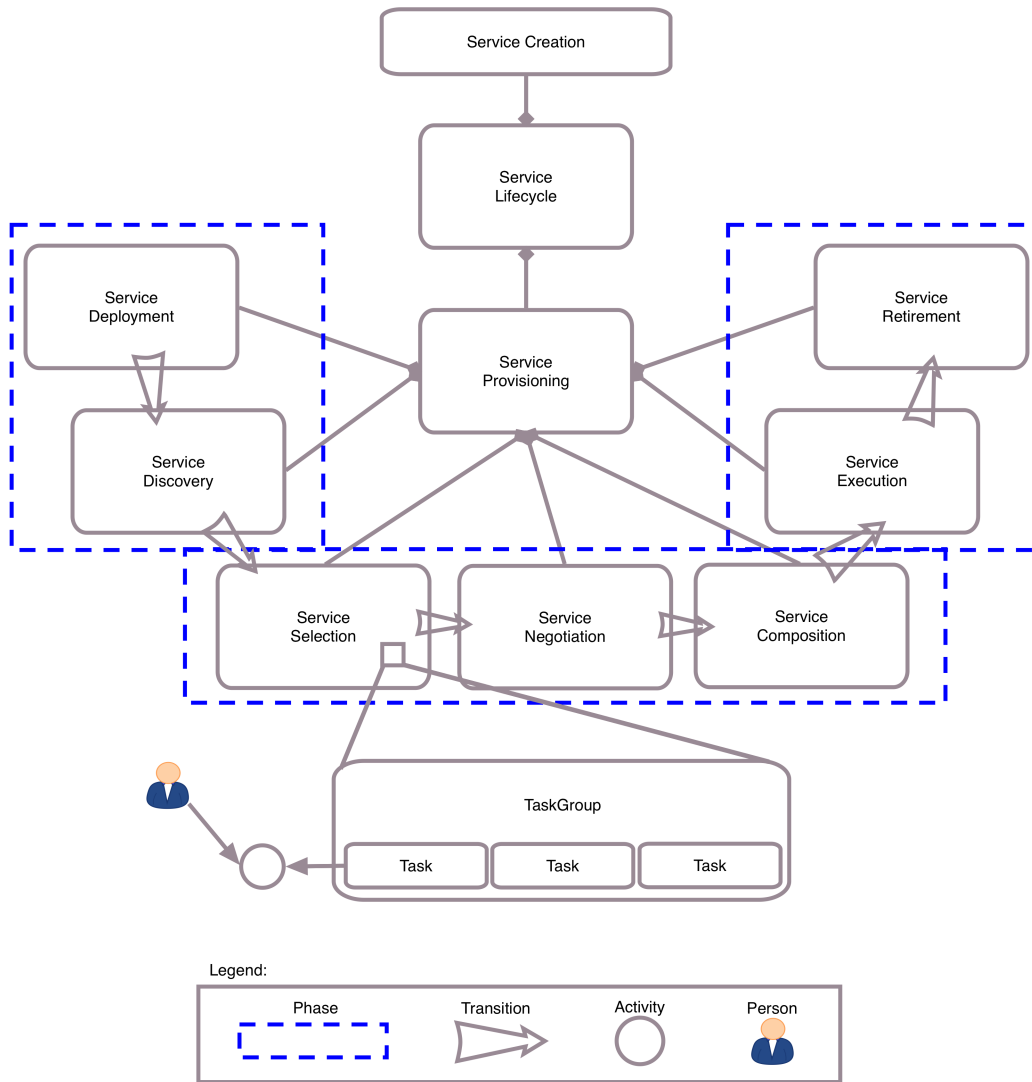


Figure 5.1: Service Lifecycle example

CHAPTER 5. SEMANTIC MODELS

Classes	
Name	Description
Activity	An Activity describes an instance of a Task.
Phase	A Phase defines one possible phase that a Service may be at a specific time
Role	A Role defines a role or a set of roles involved in a Service Phase, TaskGroup, Task or Activity.
ServiceLifecycle	A ServiceLifecycle consists in a series of phases with their respective transitions between them.
Task	A Task represents an item that must be completed in a specific time.
TaskGroup	A Task Group describes a set of Tasks grouped together.
Time	Time is useful to describe the time constraints of a Phase, Task or Activity.
Transition	A Transition describes how a transition is made between Phases. It also may describe what tasks are needed to make this Transition.

Table 5.2: Service Lifecycle Ontology Properties

5.1. SERVICE LIFECYCLE ONTOLOGY

Classes	
Name	Description
activityStatus	This property gives us information about the status of an Activity.
actualPhase	An actual Phase serves to describe the actual phase of a Service. It is important to note that a Service can only have one Phase at a specific time.
durationTime	This property gives the duration time of a specific Phase, Task or Activity.
endTime	This property gives the ending time of a specific Phase, Task or Activity.
hasPerson	This property specifies a person that belongs to a Role.
hasRole	This property specifies a role.
hasTask	With this property it is possible to describe an assignment.
hasTaskGroup	With this property it is possible to describe a set of Tasks.
isActivityCompleted	This property specify if the Activity is completed.
isTaskCompleted	This property specifies if the Task is completed.
isTaskStarted	This property specifies if the Task has started.
isTaskStoped	This property describes if the Task is on hold.
possiblePhase	A possible Phase describes the different phases that are available in a certain Service Lifecycle.
possibleTransition	A possible Transition describes all the phases that are valid or available for the next Transition.
resource	A resource is something that is processed in the Activity.
resultingPhase	A resulting Phase is useful to describe the resulting phase after a successful Transition.
startTime	This property gives the initial time of a specific Phase, Task or Activity.
taskStatus	This property gives us information about the status of a Task.

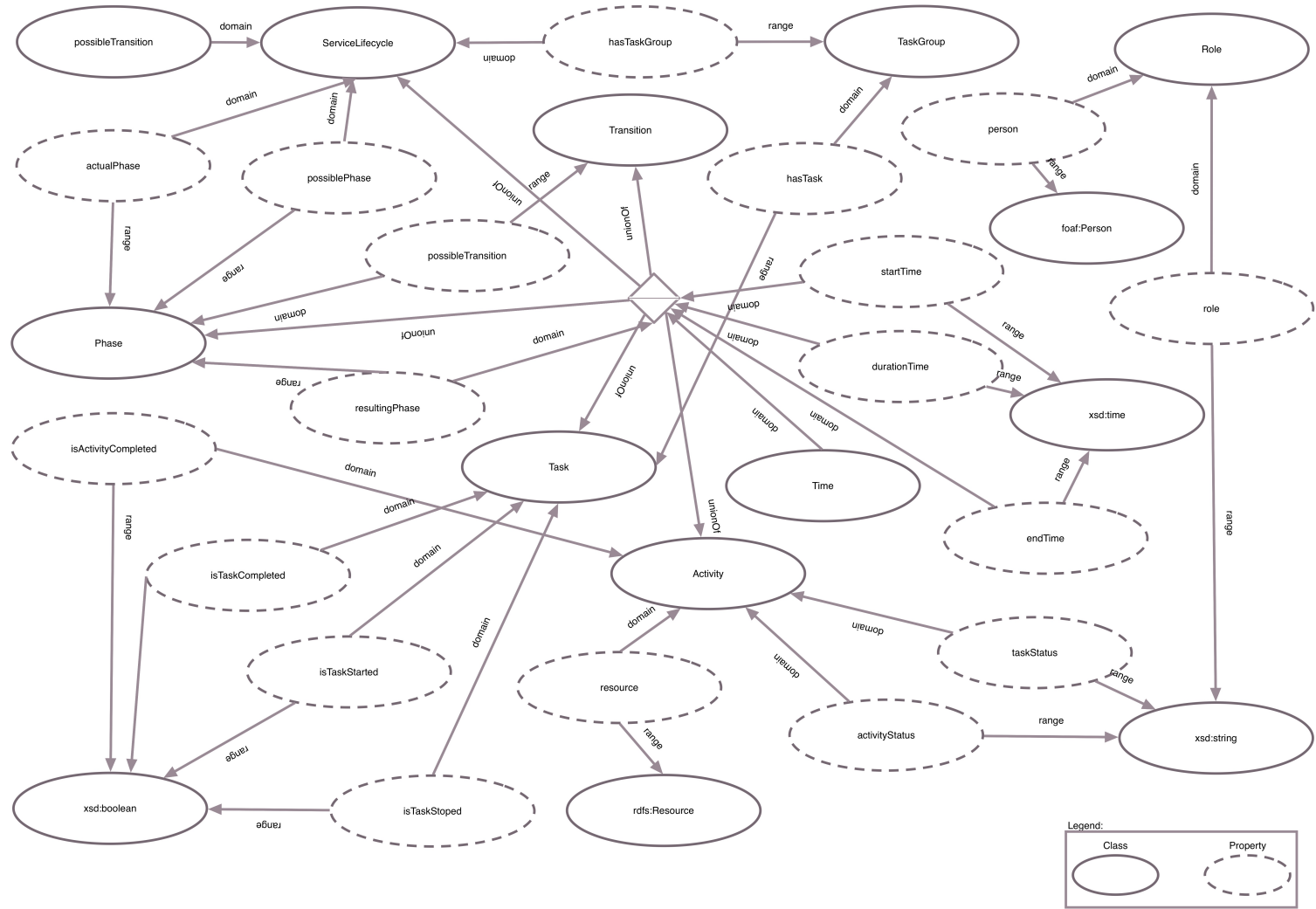


Figure 5.2: Service Lifecycle Ontology Diagram

5.2. SAPO SERVICES TAXONOMY

This initial version of the Ontology will always be subject to changes that may arise in order to improve its quality.

5.2 SAPO Services Taxonomy

The SAPO Services Taxonomy was created in order to be used together with Linked-USDL to describe some SAPO Services. This is not a requirement of the project but an way to demonstrate that Linked-USDL could easily describe them.

The full description is presented in the appendix D.

6

Service Model Mapping

6.1 Approaches to Mapping

The main purpose of this thesis was to do a study to see if it was feasible and possible to export/import between two different models without making any extension to USDL. This would allow to transfer a list of services offers through their environments, such as of development, quality and production.

As seen in the previously Chapters we initially started by analysing the SDB data scheme and the USDL model to see which differences existed. After performing the mappings we decided to use a transformation file along with C#: a programming language requested by the stakeholder. Although, we change that to a Language Integrated Query (LINQ) which allows to convert a XML document to a collection of XElement objects, “which are then queried against using the local execution engine that is provided as a part of the implementation of the standard query operator“. This enabled an easily manipulation of all entities in both data models.

When it is impossible to create a direct mapping from SDB data scheme to USDL it will be created an extra property. USDL allows us to specify a file containing technical configurations which in our case is almost all information. We decided then to mapping all the common technical and operational information when they exist in both models and make reference to the files at the same time. This will allow us to have a more complete description without having to create any extension. In cases where we are unable to do a direct mapping we decided to create an extension to be able to do the export/import.

In the next section we discussed how we established the mappings and why we made that way.

6.2 Establishing Mappings

The USDL language attempts to describe a range of services from different domains. In order to be possible they have defined a generic language who could bear the greatest number of domains. However, it is a challenge to cover all scenarios which in many cases, must be implemented new attributes.

The SDB data scheme has this problem which was only outlined with an extension to the USDL model. This extension is able to model some non-existence USDL concepts such as Projects, Schemas, Strategies, etc.

The existence of this concepts does not necessary imply that the USDL has been poorly defined but a necessity in the SDB data scheme for describing services.

In the next Section we show how the extension was made and also the necessary attributes. The SDB data scheme is not presented for correlation due to confidentiality issues.

6.2.1 General Module Information

Parameters of the package that captures the module

- **Namespace:** <http://services.sapo.pt/usdl/modules/sdbextension>;
- **Name:** sdbextension.

Figure 6.1 shows the class diagram of the package that captures the SDB data scheme concepts. This diagram also shows which associations are compositions and which ones are normal relationships.

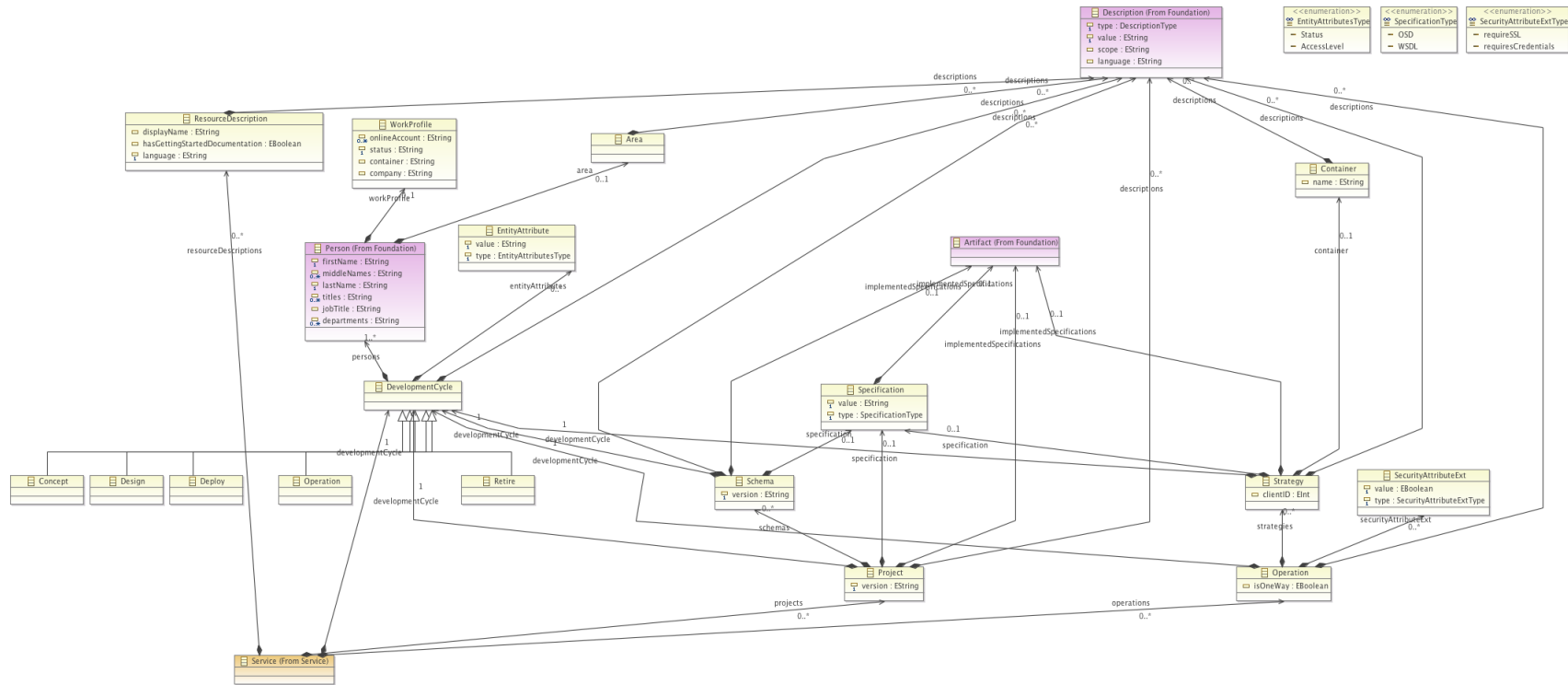


Figure 6.1: Class diagram of the package that captures the SDB data scheme concepts.

6.2.2 Module Dependencies

This section give us an overview of the concepts used in the extension. In order to better understand the concepts used, it is recommended to read the following documents:

- Service[7]
- Foundation[6]

Table 6.1 shows a quick overview of the concepts to avoid extensive readings. All information is based on USDL documents¹.

Table 6.1: overview of concepts used in SDB data scheme extension

Name	Type	Module	Description
Service	EClass	Service	A network-provisioned entity that offers that offers capabilities, which are exposed through technical interface.
Person	EClass	Foundation	This concept represents unique human legal entities.
Artifact	EClass	Foundation	A generic concept that allows to point to service metadata outside of USDL, as well arbitrary documents, files, etc.
Description	EClass	Foundation	A generic concept that provides various information elements to describe USDL objects.

6.2.3 SDB data scheme extension: Model

6.2.3.1 ResourceDescription

A Resource description is a concept that allows describing a service in a business perspective and sets if a service has initial documentation.

- ***Ecore Type:*** EClass;
- ***Interfaces:*** N/A;
- ***Superclass:*** N/A.

¹<http://www.internet-of-services.com/index.php?id=288&L=0>

6.2. ESTABLISHING MAPPINGS

Table 6.2: ResourceDescription concept

ResourceDescription			
Attributes			
Name	Type	Cardinality	Description
displayName	EString	0..1	The display name of the Resource Description of the service.
hasGettingStartedDocumentation	EBoolean	0..1	This sets the possibility of getting started documentation.
language	EString	0..1	Language used for describing USDL object. valid values: 2-letter or 3-letter language codes accordingly to ISO 639-1, ISO 639-2 or ISO 639-3.
Relations			
Name	Type	Cardinality	Description
descriptions	Description	0..*	Set of (additional) descriptive information about the Resource description, possibly in multiple natural languages

6.2.3.2 DevelopmentCycle

A DevelopmentCycle is a concept to describe the parties that are engaged in the entity process. It also describes some properties of the entity such as the status and access level.

Table 6.3: DevelopmentCycle concept

CHAPTER 6. SERVICE MODEL MAPPING

DevelopmentCycle			
Relations			
Name	Type	Cardinality	Description
persons	Person	1..*	Set of Persons that belong to a DevelopmentCycle.
entityAttributes	EntityAttributes	0..*	A Set of Attributes
descriptions	Description	0..*	A Set of (additional) descriptive information about the Resource description, possibly in multiple natural languages

6.2.3.3 WorkProfile

A WorkProfile represents additional information about a Person. Encompasses online accounts, the associated company, etc.

Table 6.4: WorkProfile concept

WorkProfile			
Attributes			
Name	Type	Cardinality	Description
onlineAccount	EString	0..*	A set of online accounts, e.g., SAPO services.
status	EString	1	The Person's status.
container	EString	0..1	The Person's container.
company	EString	0..1	The Person's company.

6.2.3.4 Area

An Area represents the specific work field of a Person. It is possible to describe in different languages.

Table 6.5: Area concept

6.2. ESTABLISHING MAPPINGS

Area			
Relations			
Name	Type	Cardinality	Description
descriptions	Description	0..*	A Set of (additional) descriptive information about the Resource description, possibly in multiple natural languages

6.2.3.5 EntityAttribute

An EntityAttribute represents the set of attributes that a specific entity have.

Table 6.6: EntityAttribute concept

grey			
Attributes			
Name	Type	Cardinality	Description
value	EString	1	The value of the status and accessLevel.
type	EntityAttributesType	1	Type denotes the nature of the Attribute.

6.2.3.6 Project

A Project is a concept that represents a plan, with involved parties and a owner.

Table 6.7: Project concept

CHAPTER 6. SERVICE MODEL MAPPING

Project			
Attributes			
Name	Type	Cardinality	Description
version	EString	1	The version number of a Project.
Relations			
Name	Type	Cardinality	Description
developmentCycle	DevelopmentCycle	1	A developmentCycle is a concept to describe the process and parties that are engaged in the entity process.
schemas	Schema	1	A set of schemas that a project may have.
specification	Specification	1	A project can have one specification.
implementedSpecifications	Artifact	1	A project may have an configuration file. The implemented-Specifications points to that file.
descriptions	Description	0..*	Set of (additional) descriptive information about the Resource description, possibly in multiple natural languages

6.2.3.7 Schema

A Schema defines the structure, and to some extent, the semantics of a document. The structure is similar to the Project with the difference that the Project have a Schema and not the other way.

Table 6.8: Schema concept

6.2. ESTABLISHING MAPPINGS

Schema			
Attributes			
Name	Type	Cardinality	Description
version	EString	1	The version number of a Project.
Relations			
Name	Type	Cardinality	Description
developmentCycle	DevelopmentCycle	1	A developmentCycle is a concept to describe the process and parties that are engaged in the entity process.
specification	Specification	1	A project can have one specification.
implementedSpecifications	Artifact	1	A project may have an configuration file. The implemented-Specifications points to that file.
descriptions	Description	0..*	Set of (additional) descriptive information about the Resource description, possibly in multiple natural languages

6.2.3.8 Specification

Table 6.9: Specification concept

CHAPTER 6. SERVICE MODEL MAPPING

Schema			
Attributes			
Name	Type	Cardinality	Description
value	EString	1	The value of the Project type.
type	SpecificationType	1	The type can be one of the available elements in SpecificationType .
Relations			
Name	Type	Cardinality	Description
implementedSpecifications	Artifact	1	A project may have an configuration file. The implemented-Specifications points to that file.

6.2.3.9 Operation

A Operation concept has parties in the same way as Service and Project. It differs in the Strategies and security elements.

Table 6.10: Operation concept

6.2. ESTABLISHING MAPPINGS

Operation			
Attributes			
Name	Type	Cardinality	Description
isOneWay	EBoolean	1	isOneWay represents the way the message is delivered.
Relations			
Name	Type	Cardinality	Description
developmentCycle	DevelopmentCycle	1	A developmentCycle is a concept to describe the process and parties that are engaged in the entity process.
specification	Specification	1	A project can have one specification.
implementedSpecifications	Artifact	1	A project may have an configuration file. The implemented-Specifications points to that file.
container	Container	1	A strategy may have a container that comprises a name and descriptions.
descriptions	Description	0..*	Set of (additional) descriptive information about the Resource description, possibly in multiple natural languages

6.2.3.10 SecurityAttributeExt

A SecurityAttributeExt is a concept that represents some of security attributes that are necessary to model the Operation concept.

Table 6.11: SecurityAttributeExt concept

SecurityAttributeExt			
Attributes			
Name	Type	Cardinality	Description
value	EBoolean	1	Represents the value that the type elements can take.
type	SecurityAttributeExtType	1	Points to the SecurityAttributeExtType. It can take two different values: requiresSSL and requiresCredentials

6.2.3.11 Container

A Container concept represents a set of a Strategy descriptions.

Table 6.12: Container concept

Container			
Attributes			
Name	Type	Cardinality	Description
name	EString	1	name represents the name of the container.
Relations			
Name	Type	Cardinality	Description
descriptions	Description	0..*	Set of (additional) descriptive information about the Resource description, possibly in multiple natural languages

EntityAttributesType

EntityAttributeType is used to specify the Status and AccessLevel of an entity.

Table 6.13: EntityAttributesType concept

6.2. ESTABLISHING MAPPINGS

EntityAttributesType		
Items		
Name	int Value	Description
Status	0	The value of the Entity type that specifies the Status.
AccessLevel	1	The value of the Entity type that specifies the AccessLevel.

6.2.3.12 SpecificationType

SpecificationType is used to specify if the file is a OSD or a WSDL.

Table 6.14: SpecificationType concept

SpecificationType		
Items		
Name	int Value	Description
OSD	0	The value of the Specification type that specifies an OSD configuration.
WSDL	1	The value of the Specification type that specifies a WSDL configuration.

6.2.3.13 SecurityAttributeExtType

SecurityAttributeExtType is used to specify the requireSSL and the requiresCredentials.

Table 6.15: SecurityAttributeExtType concept

SecurityAttributeExtType		
Items		
Name	int Value	Description
requireSSL	0	The value of the Security type that specifies if a Operation requiresSSL.
requiresCredentials	1	The value of the Security type that specifies if a Operation requiresCredentials.

6.3 Export/Import Application

In this Chapter we will describe how our Export/Import was implemented, what decisions were made during this process and the technologies used. We will start with a brief description of the technologies used as well as the reason for these choices.

In the next Section we will explain in detail the configurations and the implementation process.

6.3.1 System technologies

When this project started was decided that we should use, as requirement, Web services and the C# programming language. We could also use some libraries and programming languages that might be necessary to facilitate the task.

We decided to use a Web interface in Active Server Pages (ASP) and also Javascript for simplifying the interface and do some asynchronous calls to the server (AJAX). As mention before the main language is C# combined with Web services which allow a communication between two different devices through the World Wide Web. The last technology used was Linq to XML which is an alternative much easier to use than a XSLT file and adds native data querying capabilities to .NET languages.

6.3.2 Implementation

The first page presented to the user is the login page. This only exists because the deploy is on a public domain where we decided to protect the information from non-authentic users. The process is simple: we have a database

6.3. EXPORT/IMPORT APPLICATION

on server side in MySQL with only one table and a user. When the user attempts to make a login this information is send to the server and a validation may occur. If the output is valid the user will have access to the other pages.

The export/import page is much more complex than the previous page. We use one API from HTML5 called File API in order to let the user open remote files from the computer. We have a mechanism that allows us to know which type of Export we will do. This means that we only need one kind of Export button. After this selecting is made it will be presented the Export.xml information in a special Textbox1 which uses a plugin called codemirror. This plugin allow us to indent the information according to the selected file and giving us lot of editing customizations. All the remaining files will be loaded in memory until the user decides to click the Export button. When this happens it will analyse the Export.xml to see how many external files exist in that file and check the files loaded to see if they are the correct ones. Next, it will call a remote Web service asynchronous through AJAX. We are calling this Web service this way because we need to update only one element in the page which is the Textbox2.

Internally, the Web service will start to make all the necessary changes through the Linq to XML. We analyse line by line to see what kind of mappings it should be addressed. This process depends of the file received to see if we are making an Export of the type USDL or SDB data scheme. When this process is finished we response through AJAX to the user interface (Texbox2). This allow us to see the resulting file and if we want it, to make changes before downloading the file. We have this option always available to the user through Textbox1 and Textbox2.

This is the full process of the Export/Import application. There are some particular aspects on the code that we will not address in this explanation.

6.3.3 Configurations

There are few configurations for this project. All the Web services were on the same machine so it was not necessary to change the Web.config file. In the next Sections we will show the few configurations for the database and the asynchronous calls that we perform on Web services for the Export/Import application.

6.3.3.1 Database

The database is only for using in the login and as such it was a fairly straightforward process. We only create one Table and set the user and pass-

CHAPTER 6. SERVICE MODEL MAPPING

word to sunset. In Figure 6.2 we show how the connection to the Web site is created.

```
1 CREATE TABLE Persons
2 (
3 id int,
4 username varchar(20),
5 password varchar(20),
6 );
7
8 INSERT INTO Persons
9 VALUES (1, 'sunset', 'sunset');
```

Listing 6.1: Database Table creation

```
1 SqlConnection connection = new SqlConnection("workstation id=sunsetsql.mssql.
somee.com;packet size=4096;user id=redeagle18_SQLLogin_1;pwd=i9gr9kevnt;
data source=sunsetsql.mssql.somee.com;persist security info=False;initial
catalog=sunsetsql");
```

Listing 6.2: Database configurations

6.3.3.2 AJAX

In Figure 6.3 is shown how we are making the calls to the Web service. We have two different types of extension files where according to each one it will call different methods.

```
1
2 if (ext == 'xml') {
3     $.ajax({
4         dataType: "json",
5         type: "POST",
6         url: "WebService1.asmx/convertToUSD",
7         contentType: "application/json; charset=utf-8",
8         //data: "{XmlPath:'"+ str +"', ListFiles: '"+JSON.stringify(files) +"'",
9         data: "{XmlPath:'" + str + "', ListFiles: '" + JSON.stringify(files) + "
            '}",
10
11         success: function (data) {
12
13             if (data.d != null) {
14                 editor2.setValue(data.d);
15                 document.getElementById("download").style.visibility = "visible";
16             }
17         },
18         error: function () {
19             alert("Invalid format!!!");
20         }
21     });
22
23     });
24     return false;
25 }
26 if (ext == 'rdf') {
27     $.ajax({
```

```

28     dataType: "json",
29     type: "POST",
30     url: "WebService1.aspx/convertToXML",
31     contentType: "application/json",
32     data: "{usdl:'" + str + "'}",
33     success: function (data) {
34
35         if (data.d != null) {
36             editor2.setValue(data.d);
37             document.getElementById("download").style.visibility = "visible";
38
39         }
40     },
41     error: function () {
42         alert("Invalid Format!!!");
43
44     }
45
46 });
47 return false;
48 }

```

Listing 6.3: AJAX calls

6.4 Evaluation

The Export/Import is an application that depends largely on the mappings. There are a few things that we could analysed besides that and they are described on the requirements document. This application as a simple concept and as such, it also as a simple evaluation.

We divided the tests accordingly to the requirements document which are: Functional, Non-functional, Usability, Reliability, Design, Implementation and Interface tests.

6.4.1 Test Plan

Our Test plan do not differentiate the user since the application does not have user profiles. There are few hardware and software requirements for this Test plan because nowadays every machine as a browser and a internet connection.

As previously mentioned we will tested some requirements from the requirement's document. It will be tested the 01, 02, 03, 04, 05, 06, 07 from Functional list, 01, 02, 03 from the Usability list, 01, 02 from the Reliability list and 01 from Design list as seen in Appendix B.

6.4.2 Test Case

In this Section we presented the tests that are necessary to see the quality of the application:

Functional Tests

- **01: Choose remote files** This test will consist in choosing two different files from our local machine ;
- **02: Choose multiple files** This test is similar to the previous with the difference that we will choose various files at the same time ;
- **03: Restriction on file selection** In this test we will pick different files with different extensions to see if the system could load the file ;
- **04: File validation before export** We will load different files and then we will try to do the Export ;
- **05: Download the resulting file** In this test we will download the resulting file even if it is presented as null ;
- **06: Possibility of making changes on the fly** Here we will try to change the text in both Textboxes and see if it was possible to change that ;
- **07: Files automatically formatted** We will test if on the loading process the file is automatically formatted. We will try different files .

Usability

- **01: The information loaded is presented according to their language** We will try different files to see if the loaded file is presented according to their language ;
- **02: The resulting information is presented according to their language** We will try to make the Export in order to see if the resulting file is presented according to their language ;
- **03: Errors and warnings** To test this we will try to load unsupported files and trying to do the Export with invalid files .

Reliability

- **01: Correct Export from SDB data scheme to USDL** We will compare the Export resulting file with the mappings that were made manually during the study process ;
- **02: Correct Export from USDL to SDB data scheme** This test is equal to the previous one, 01 .

Design

- **01: Login protection** In this test we need to test a correct username and password and also wrong usernames and passwords .

6.4.3 Results

In this Section are described succinctly the results as seen in Table 3.3

Table 6.16: Functional Tests list

ID	Name	Results
01	Choose remote files	It was possible to open different files from our local machine.
02	Choose multiple files	The application also supports loading different files at the same time.
03	Restriction on file selection	Only XML files and RDF files are supported.
04	File validation before export	If the Export.xml file is different from the XSD the application will not allow the Export
05	Download the resulting file	It is possible to download the resulting file and also a empty file.
06	Possibility of making changes on the fly	The application allows the user to change all the text in both Textboxes.
07	Files automatically formatted	When the user loads a file it is formatted automatically according to the language.

Table 6.17: Usability Tests list

ID	Name	Results
01	The information loaded is presented according to their language	Every time we load a file that is valid the application sets the correct language.
02	The resulting information is presented according to their language	When the Export is done the application will presented the resulting file accordingly to their language.
03	Errors and warnings	There are some warnings: when the user tries to load an invalid file; when the user loads a file with the correct extension but its invalid to do the Export.

CHAPTER 6. SERVICE MODEL MAPPING

Table 6.18: Reliability Tests list

ID	Name	Results
01	Correct Export from SDB data scheme to USDL	The Export to USDL is identical to the previous analyse made.
02	Correct Export from USDL to SDB data scheme	The Export to SDB data scheme is identical to the previous analyse made.

Table 6.19: Design Tests list

ID	Name	Results
01	Login protection	The Login works correctly with valid and invalid users.

6.4.3.1 Limitations and Improvements

We could improve in some aspects:

- ***A list of files*** We could have a Textbox where we select the files we wanted to use in the Export process;
- ***Visual mappings*** This is a bit complex but it would be very interesting to do visual mappings. The user could open different files and drag elements from one file to another in order to do the mappings.

There are also some limitations:

- ***Doesn't work well for large files*** We have to Export one service each time;
- ***Different Export formats*** It would be very interesting if this application could Export/Import to different models.

7

Conclusion

7.1 Summary

The main goal of this Thesis was to create an Export/Import application that could use the USDL model to Export and Import all the SAPO services available on their marketplace to a specific service description language. However, it was necessary to create an extension which pulled some advantages that are inherent to the USDL model as we will explain in the next paragraph.

During this process we analysed different technologies and models enabling us to draw some interesting conclusions. The main purpose of USDL goes beyond the bindings, endpoints, operations, parameters, etc, that are described by the WSDL and their semantic part. It allows to fully describe through only one description language all the concepts related to business services, electronic services, etc. This language is also relevant if we want to integrate different product offers through multiple providers which has the advantage of being a well-defined and closed language. Even if this is not a need it allows to describe services or a service offer in a standard way, such as pricing models, security, SLA's, etc. With this in mind one of the main reasons of the Export/Import was to move service offers through their different environments such as production, quality and development. As more applications appear to the market it certainly will benefit the interoperability of the USDL model. In order to finish and although is not part of the work, Linked-USDL could bring many advantages to the semantic search in which will be very interesting to do a search in a service catalog.

7.2 Future Work

We think that the application could be improved in many ways. We could add more model languages for the Export or an option for adding visual mappings which would be very interesting. We can also think in semantic terms and started to add more functions to this. Linked-USDL could improve in many cases the search in comparison to others mechanisms which would be very interesting to add this to a service offering.

Both models have their advantages and disadvantages but both could fully describe all kind of services, such as business services and electronic services.

Bibliography

- [1] Capturing architectural requirements.
- [2] Fmc quick introduction.
- [3] Oecd internet economy outlook.
- [4] Service delivery broker.
- [5] M. Armbrust, A. Fox, A. D. J. Rean Griffith, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, , and M. Zaharia. Above the clouds: A berkeley view of cloud computing.
- [6] A. Barros, U. Kylau, and D. Oberle. Unified service description language 3.0 foundation.
- [7] A. Barros, U. Kylau, and D. Oberle. Unified service description language 3.0 service.
- [8] M. Bichler and K.-J. Lin. Service-oriented computing.
- [9] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*.
- [10] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard. Web services architecture.
- [11] R. Buyya, C. S. Yeo, , and S. Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities.
- [12] J. Cardoso. *Modeling Services with α -USDL*.
- [13] J. Cardoso. Towards a unified service description language for the internet of services: Requirements and first developments.
- [14] J. Cardoso, K. Voigt, , and M. Winkler. Service engineering for the internet of services.
- [15] J. Cardoso, M. Winkler, and K. Voigt. A service description language for the internet of services.
- [16] S.-Y. Choi and A. B. Whinston. Benefits and requirements for interoperability in the electronic marketplace.

BIBLIOGRAPHY

- [17] I. Churakova and R. Mikhranova. Software as a service: Study and analysis of saas business model and innovation ecosystems.
- [18] D. Clegg and R. Barker. *Case Method Fast-Track: A Rad Approach*.
- [19] C. M. Cleland and D. M. Gomez. Prospects for caricom services exports in information and communications technology: Trade and investment issues.
- [20] R. J. Cooper. Measuring the impact of innovations in public it infrastructure on the standard of living in oecd economies.
- [21] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana. Unraveling the web services web: An introduction to soap, wsdl, and uddi.
- [22] T. Erl. Soa principles of service design.
- [23] D. Fensel and C. Bussler. The web service modeling framework wsmf.
- [24] D. Gago and L. Rubalcaba. Innovation and ict in service firms: Towards a multidimensional approach for impact assessment.
- [25] D. Gerundino and R. Weissinger. *Economic benefits of standards - International case studies Volume 2*.
- [26] B. Hayes. News cloud computing.
- [27] F. Heberie. Comparison of service offerings in the future internet, spanning over multiple providers and stores.
- [28] T. P. Hill. On goods and services.
- [29] C. W. Kian. The design and development of an e-marketing framework for the asian b2b marketplace.
- [30] A. Knopfel, B. Grone, and P. Tabeling. *Fundamental Modeling Concepts: Effective Communication of IT Systems*.
- [31] T. Kretschmer. Information and communication technologies and productivity growth: A survey of the literature.
- [32] A. Lenk, M. Klems, J. Nimis, S. Tai, and T. Sandholm. What's inside the cloud? an architectural map of the cloud landscape.
- [33] L. Li and I. Horrocks. A software framework for matchmaking based on semantic web technology.

- [34] P. P. Maglio, S. L. Vargo, N. Caswell, and J. Spohrer. The service system is the basic abstraction of service science.
- [35] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara. Owl-s: Semantic markup for web services.
- [36] D. L. McGuinness and F. van Harmelen. Owl web ontology language overview.
- [37] P. Mell and T. Grance. The nist definition of cloud computing (draft).
- [38] H. R. Montahari-Nezhad, B. Stephenson, and S. Singhal. Outsourcing business to cloud computing services: Opportunities and challenges.
- [39] A. Plepys. The grey side of ict.
- [40] M. Pohjola. The new economy: Facts, impacts and policies.
- [41] C. Reimsbach-Kounatze and C. S. Vallejo. Ict skills and employment: New competences and jobs for a greener and smarter economy.
- [42] B. P. Rimal, E. Choi, and I. Lumb. A taxonomy and survey of cloud computing systems.
- [43] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, and D. Fensel. Web service modeling ontology.
- [44] K. Sapprasert. The impact of ict on the growth of the service industry.
- [45] A. P. Sheth, K. Gomadam, and A. Ranabahu. Semantics enhanced services: Meteor-s, sawsdl and sa-rest.
- [46] I. Sommerville. *Software Engineering*.
- [47] I. Sommerville. *Software Engineering Ninth Edition*.
- [48] J. Spohrer, P. P. Maglio, J. Bailey, and D. Gruhl. Steps toward a science of service systems.
- [49] P. Stryszowski. The impact of the internet in oecd countries.
- [50] H. van der Wiel, G. van Leeuwen, and T. Hempell. Ict, innovation and business performance in services: Evidence for germany and the netherlands.

BIBLIOGRAPHY

- [51] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. A break in the clouds: Towards a cloud definition.
- [52] T. Vitvar, J. Kopecky, and D. Fensel. Wsmo-lite: Lightweight semantic descriptions for services on the web.
- [53] L. Wang and G. V. Laszewski. Scientific cloud computing: Early definition and experience.
- [54] D. Weller and B. Woodcock. Internet traffic exchange.

Appendices

A

GIS Use Case

In this Chapter we presented GIS, a service available at SAPO Services, which was modelled through Linked-USDL. The following sections are organized in the respective form: the Section A.2 shows the reason behind the choice of GIS service. The Section A.3 present the technologies used while modelling this service. The Section A.4 gives the methodology followed while were modelling. The last Section A.5 shows Linked-USDL Service Modeling process and it gives a description of the service vocabulary.

A.1 Introduction

As previously mention, the service being modelled is one of the available in SAPO Services. GIS Studio¹ is a web application in SaaS model, that works in any browser and operation system and doesn't require installation of any software on the client.

A.2 Why GIS?

The initial objective was to describe a service using Linked-USDL so that we can gain a better understanding of the modelling process. The second objective was that this service must belong to SAPO Services² and it must be one of the services which has more information. With this being said, GIS service is within the requirements.

¹<https://store.services.sapo.pt/en/Catalog/maps/gis-studio>

²<https://store.services.sapo.pt/en/>

APPENDIX A. GIS USE CASE

GIS Studio

Web application (SaaS - Software as a Service) that allows you to manage geographic content: Points of Interest (POI), statistics, itinerary, among others.

[Buy product](#)



Overview

Pricing

Coverage

Technical Documentation

GIS Studio is a web application in SaaS (Software as a Service) model, that works in any browser and operating system and doesn't require installation of any software on the client or server.

The GIS Studio allow you to manage your own data, maps and geographic information, in a system based on standard technologies, open, with mapping, analysis, collaboration and editing tools.

Features

The application includes the following modules:

- ▶ **Points of Interest** - Allows you to search, create and edit points of interest and its details: address, telephone, e-mail, website, foto, etc;
- ▶ **Itineraries** - Adding, edit, and delete itineraries as well as availability of these on your site;
- ▶ **Supervision** - Supervise (accept or refuse) changes made to your points of interest by your or someone of your entity;
- ▶ **Data Import** - Importation of files in kml or shapefile format, conversion of several or different coordinate systems allowing a greater compatibility between the various existing formats;
- ▶ **Statistics** - Module which shows the statistical indicators about quantitative data on various economic and social sectors;
- ▶ **Entities** - Management of information related to user permissions of your entity.

Figure A.1: GIS Studio Overview

A.3 Technologies Used

The technologies used are based in the Semantic Web³ and Linked Data [?]. All the code was written in Turtle⁴ which is a more "human friendly" way of write RDF⁵ triples.

A.4 Methodology

The methodology used was the one followed in [27], where the author describes one way of modelling a service in Linked-USDL.

Before starting to model the service, a vocabulary was created with all the information obtained on GIS Studio website. This is explained in section A.5.6. After this, the modelling is done with all the information available about the service. Unfortunately some was not available, as it was the case of SLA's. In the end, two documents are produced, one with the Service instantiation and the other with the Service vocabulary. Both are Turtle files.

³<http://www.w3.org/standards/semanticweb/>


⁴<http://www.w3.org/TR/turtle/>

⁵<http://www.w3.org/RDF/>

A.5. LINKED-USDL SERVICE MODELING

GIS Studio
Web application (SaaS - Software as a Service) that allows you to manage geographic content: Points of Interest (POI), statistics, itinerary, among others.

[Buy product](#)



Overview **Pricing** Coverage Technical Documentation

Prices for 1 Month

Module	
Base Module	€184.50
Each Additional Module	€61.50

All prices shown include VAT at the legal rate.

The base module includes:

- ▶ Entity management module (with the possibility of inserting up to 3 email accounts);
- ▶ POIs management module;
- ▶ Importing files in kml and shapefile module.

For other solutions contact our team using the [contacts page](#).

Figure A.2: GIS Studio Pricing

A.5 Linked-USDL Service Modeling

In this section will be explained how GIS instantiation was made. Note that Linked-USDL is still under development at the time of writing.

A.5.1 Vocabulary Prefixes

This section shows the most important prefixes for the vocabulary used in the Service instantiation.

Thanks to Linked-DATA⁶ is possible to refer to the knowledge already collected. This is a tremendous advantage because it is not necessary to create these concepts again. In Listing A.1, it is shown exactly that. The prefixes in Line 4,5,6 and 7 are the Linked-USDL modules. The rest presented, describe different concepts not available in Linked-USDL.

*GoodRelations*⁷ and *SKOS*⁸ will be most used prefixes through service description. The first, describes prices, products, and other elements and the second provides a standard way to represent knowledge organization systems.

```
1 @base <http://rdfs.genssiz.org/GIS> .
2 @prefix : <http://rdfs.genssiz.org/GIS#> .
3 @prefix bime: <http://rdfs.genssiz.org/GIS_Vocabulary_01#> .
4 @prefix usdl: <http://www.linked-usdl.org/ns/usdl-core#> .
5 @prefix legal: <http://www.linked-usdl.org/ns/usdl-legal#> .
```

⁶<http://www.w3.org/standards/semanticweb/data>

⁷<http://www.heppnetz.de/ontologies/goodrelations/v1>

⁸<http://www.w3.org/2009/08/skos-reference/skos.html>

APPENDIX A. GIS USE CASE

```
6 @prefix price: <http://www.linked-usdl.org/ns/usdl-pricing#> .
7 @prefix sla: <http://www.linked-usdl.org/ns/usdl-sla#> .
8 @prefix skos: <http://www.w3.org/2004/02/skos/core#> .
9 @prefix dcterms: <http://purl.org/dc/terms/> .
10 @prefix owl: <http://www.w3.org/2002/07/owl#> .
11 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
12 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
13 @prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

Listing A.1: Vocabulary Prefixes

A.5.2 Service Instance

As shown in Listing A.2, the service instantiation uses the following vocabularies: *USDL-Core*, *dcterms* and *GoodRelations*.

In line 2, as we can see, the attribute *title* of the *dcterms* vocabulary is used, and this means that the Service has a title. The next line uses the *USDL-Core* vocabulary to define the Service has a SaaS. In line 4 its used the attribute *hasProvider* from the *USDL-Core* vocabulary to define that *GISprovider* is the Service Provider. In the next line is added the legal terms, as we can see, trough the attribute *hasLegalCondition*. Listing A.2 in line 6, shows us the *GoodRelations* vocabulary. It uses the *qualitativeProductOrServiceProperty* attribute to describe the concepts defined on the service vocabulary, as explained in section A.5.6. This service has no quantitative properties and as such will not be addressed.

```
1 <#GIS_service> a usdl:Service ;
2   dcterms:title "GIS Studio - Web application (SaaS - Software as a Service)
   that allows you to manage geographic content: Points of Interest (POI),
   statistics, itinerary, among others"@en ;
3   usdl:hasClassification :SaaS ;
4   usdl:hasProvider :GIS_provider ;
5   usdl:hasLegalCondition :GIS_legal ;
6   gr:qualitativeProductOrServiceProperty
7     gis:General_Features ,
8     gis:Advantages ,
9     gis:Security_Features ,
10    gis:Customer_Support ,
11    gis:Requirements .
```

Listing A.2: Linked-USDL GIS Instance

Listing A.3 and A.4 shows the SaaS concept and the GIS provider.

```
1 :SaaS a rdfs:Class, skos:Concept ;
2   dcterms:description "Service model of a SaaS" .
```

Listing A.3: SaaS Concept

```
1 :GIS_provider a gr:BusinessEntity ;
2   foaf:name "SAPO Services" ;
```

A.5. LINKED-USDL SERVICE MODELING

```
3 foaf:homepage <https://store.services.sapo.pt/en/> ;
```

Listing A.4: GIS Provider

A.5.3 Legal

As Listing A.5 shows in line 4, there is only one attribute, *hasClause*. This attribute has four clauses, all them pointing to a URL where the information resides.

```
1 :GIS_legal a legal:TermsAndConditions ;
2 dcterms:title "GIS legal statements"@en ;
3 dcterms:description "GIS legal statements are accessible at 'https://store.
  services.sapo.pt/en/Catalog/maps/gis-studio/coverage'"@en ;
4 legal:hasClause
5 [ a legal:Clause ;
6   legal:name "Terms of Use" ;
7   legal:text "https://store.services.sapo.pt/en/Catalog/maps/gis-studio/
  coverage"@en ] ,
8 [ a legal:Clause ;
9   legal:name "General Terms and Conditions of use for SAPO Services" ;
10  legal:text "https://store.services.sapo.pt/en/TermsAndConditions"@en ] ,
11 [ a legal:Clause ;
12  legal:name "Terms of use of SAPO Maps" ;
13  legal:text "http://seguranca.sapo.pt/termosdeutilizacao/sapo_mapas.html"@
  en ] ,
14 [ a legal:Clause ;
15  legal:name "Privacy policy of SAPO portal" ;
16  legal:text "http://seguranca.sapo.pt/politicadeprivacidade.html"@en ] .
```

Listing A.5: GIS Pricing

A.5.4 SLA

The service website did not have any information about Service Level Agreement, so it was impossible to model this part of the service.

A.5.5 Pricing

The pricing is one of the most important part of a service. First its necessary to define the service offerings.

A.5.5.1 Service Offering

Listing A.6 shows the definition of the GIS offering. As we can see, in line 4 *includes* attribute is specifying that this offering is from the Service described

APPENDIX A. GIS USE CASE

above in Section A.5.2. The next line describes the different plans that this service has. In this case, it has three different price plans.

```
1 :GIS_offering a usdl:ServiceOffering ;
2   dcterms:title "GIS instance"@en ;
3   dcterms:description "Offering for a GIS instance."@en ;
4   usdl:includes <#GIS_service> ;
5   usdl:hasPricePlan
6     :Pricing_1_month ,
7     :Pricing_6_month ,
8     :Pricing_12_month .
```

Listing A.6: GIS Offering

A.5.5.2 Price Plans

GIS Studio has three different Price Plans. Each plan, has a defined value for his use time. Inside this, the user can choose different modules to add new Service features at a single unit cost per module.

Listing A.7 show us the GIS price plan. There are two concepts that have to be mentioned: *hasContractDuration* where is defined the value and measure of the contract duration and *hasPriceComponent* which will have the description of the amount that has to be paid during this cycle and which service components are included in the price plan.

```
1 :Pricing_1_month a price:PricePlan ;
2   dcterms:title "Prices for 1 Month"@en ;
3   dcterms:description "The base module includes:Entity management module (with
4     the possibility of inserting up to 3 email accounts);POIs management
5     module;Importing files in kml and shapefile module."@en ;
6   price:hasContractDuration
7     [ a gr:QuantitativeValue ;
8       gr:hasValueInteger "1" ;
9       gr:hasUnitOfMeasurement "MON" ] ;
10  price:hasPriceComponent
11    :Price_Component_1_month ,
12    :Price_module_Itineraries_1_month ,
13    :Price_module_Statics_1_month .
```

Listing A.7: GIS Price Plan - 1 Month

A.5.5.3 Price Components

The last step is to define the different *PriceComponent* used in Listing A.7, line 8.

```
1 :Price_Component_1_month a price:PriceComponent ;
2   dcterms:title "Price for 1 Month"@en ;
3   dcterms:description "Price for the 1 Month Base Module."@en ;
4   price:isLinkedTo
```

A.5. LINKED-USDL SERVICE MODELING

Listing A.8: GIS Price Component - Part 1

The following examples, are the description of the several features of the service.

Note that all the next lines from Listing A.9 start with the word *gis*. This means that they are concepts defined in the service vocabulary A.5.6.

General Features

In Listing A.9 we describe the general features included in this service. Is appropriate to mention that all price plans have the same features.

```
1 #general Features
2 gis:Operations_POI ,
3 gis:Category_lists ,
4 gis:Sets_of_POI ,
5 gis:Coordinates ,
6 gis:Map_views ,
7 gis:Data_import ,
8 gis:Manage_entities ,
```

Listing A.9: GIS Price Component - Part 2 - General Features)

Listing A.10 shows the definition of the *General Features* group. In line 4 is used the attribute *narrower*, to define that all the below concepts are general features of the service.

```
1 :General_Features a rdfs:Class , skos:Concept ;
2 rdfs:subClassOf gr:QualitativeValue ;
3 skos:prefLabel "General Features"@en ;
4 skos:narrower
5 :Operations_POI ,
6 :Category_lists ,
7 :Sets_of_POI ,
8 :Coordinates ,
9 :Map_views ,
10 :Itineraries ,
11 :Data_import ,
12 :Statics ,
13 :Manage_entities .
```

Listing A.10: GIS Service Features - General Features

In Listing A.11, line 2, we are saying that this concept is a subclass of the *GoodRelations:Qualitative Value*.

In line 4 we specify the inheritance to the *General Features* class by using the *skos:broader* attribute.

```
1 :Operations_POI a rdfs:Class , skos:Concept ;
2 rdfs:subClassOf gr:QualitativeValue ;
```

APPENDIX A. GIS USE CASE

```
3 skos:prefLabel "Allows you to search, create and edit points of interest"@en
  ;
4 skos:broader :General_Features .
```

Listing A.11: GIS Service Features: POI operations

Advantages

This one describes the advantages of GIS service.

```
1 #Advantages
2 gis:Colaboration_between_teams ,
3 gis:Easy_to_use ,
4 gis:Without_extra_expenses ,
```

Listing A.12: GIS Price Component - Part 3 - Advantages)

As explained before, Listing A.13 shows the definition of the *Advantage Features* group. In line 4 is used the attribute *narrower*, to define that all the below concepts are general features of the service.

```
1 :Advantages a rdfs:Class , skos:Concept ;
2 rdfs:subClassOf gr:QualitativeValue ;
3 skos:prefLabel "Advantages"@en ;
4 skos:narrower
5   :Colaboration_between_teams ,
6   :Easy_to_use ,
7   :Without_extra_expenses .
```

Listing A.13: GIS Service Features - Advantages Features

In Listing A.14, line 2, we are saying that this concept is a subclass of the *GoodRelations:Qualitative Value*.

In line 4 we specify the inheritance to the *Advantages* class by using the *skos:broader* attribute.

```
1 :Colaboration_between_teams a rdfs:Class , skos:Concept ;
2 rdfs:subClassOf gr:QualitativeValue ;
3 skos:prefLabel "The different departments of your organization collaborate
  simultaneously on the data management"@en ;
4 skos:broader :Advantages .
```

Listing A.14: GIS Service Features - Colaboration between teams

Security Features

A.5. LINKED-USDL SERVICE MODELING

In this example we list the security features. This features are those responsible for ensuring data confidentiality and connection security.

```
1 #Security Features
2 gis:Sending_receiveing_data_HTTPS ,
3 gis:Encrypted_signed_data ,
4 gis:Permissions_user ,
```

Listing A.15: GIS Price Component - Part 4 - Security)

As explained before, Listing A.15 shows the definition of the *Security Features* group. In line 4 is used the attribute *narrower*, to define that all the below concepts are general features of the service.

```
1 :Security_Features a rdfs:Class , skos:Concept ;
2 rdfs:subClassOf gr:QualitativeValue ;
3 skos:prefLabel "Security Features"@en ;
4 skos:narrower
5 :Sending_receiveing_data_HTTPS ,
6 :Encrypted_signed_data ,
7 :Permissions_user .
```

Listing A.16: GIS Service Features - Security Features

In Listing A.17, line 2, we are saying that this concept is a subclass of the *GoodRelations:QualitativeValue*. In line 4 we specify the inheritance to the *Advantages* class by using the *skos:broader* attribute.

```
1 :Sending_receiveing_data_HTTPS a rdfs:Class , skos:Concept ;
2 rdfs:subClassOf gr:QualitativeValue ;
3 skos:prefLabel "he process of sending and receiving data between the
  application and the service is done through HTTPS protocol"@en ;
4 skos:broader :Security_Features .
```

Listing A.17: GIS Service Features - Sending Receiving data HTTPS

Customer Support

The customer support provides support mechanisms for the user, like chat rooms, blogs, mail support, forums and so on. In this case, there is *SOS Support*, *Mail Support* and *Blog Maps*.

```
1 #Customer Support
2 gis:SOS_support ,
3 gis:Mail_support ,
4 gis:Blog_maps ,
```

APPENDIX A. GIS USE CASE

Listing A.18: GIS Price Component - Part 4 - Customer Support)

Listing A.19 shows the definition of the *Security Features* group. In line 4 is used the attribute *narrower*, to define that all the below concepts are general features of the service.

```
1 :Customer_support a rdfs:Class , skos:Concept ;
2   rdfs:subClassOf gr:QualitativeValue ;
3   skos:prefLabel "Customer Support"@en ;
4   skos:narrower
5     :SOS_support ,
6     :Mail_support ,
7     :Blog_maps .
```

Listing A.19: GIS Service Features - Customer Support

In Listing A.20, line 2, we are saying that this concept is a subclass of the *GoodRelations:Qualitative Value*. In line 4 we specify the inheritance to the *Advantages* class by using the *skos:broader* attribute.

```
1 :SOS_support a rdfs:Class , skos:Concept ;
2   rdfs:subClassOf gr:QualitativeValue ;
3   skos:prefLabel "trata-se do apoio ao cliente SAPO, disponivel para os
4     utilizadores darem feedback ou pedirem ajuda"@pt ;
5   skos:broader :Customer_support .
```

Listing A.20: GIS Service Features - SOS Support

Service Requirements

The Service requirements are the necessary hardware and software that is needed so that the user can use the Service. In this case it is necessary a *Computer browser* and a *Mail account in SAPO Network*.

```
1 #Service Requirements
2 gis:Computer_browser ,
3 gis:Mail_account_in_SAPO_network ;
```

Listing A.21: GIS Price Component - Part 5 - Service Requirements)

Listing A.21 shows the definition of the *Requirements* group. In line 4 is used the attribute *narrower*, to define that all the below concepts are general features of the service.

```
1 :Service_requirements a rdfs:Class , skos:Concept ;
2   rdfs:subClassOf gr:QualitativeValue ;
3   skos:prefLabel "Service Requirements"@en ;
4   skos:narrower
```

A.5. LINKED-USDL SERVICE MODELING

```
5 :Computer_browser ,
6 :Mail_account_in_SAPO_network .
```

Listing A.22: GIS Service Features - Requirements

In Listing A.23, line 2, we are saying that this concept is a subclass of the *GoodRelations:QualitativeValue*.

In line 4 we specify the inheritance to the *Service_requirements* class by using the *skos:broader* attribute.

```
1 :Computer_browser a rdfs:Class , skos:Concept ;
2   rdfs:subClassOf gr:QualitativeValue ;
3   skos:prefLabel "O utilizador devera possuir um computador com um browser e
   acesso a Internet"@pt ;
4   skos:note "http://mapas.blogs.sapo.pt/" ;
5   skos:broader :Service_requirements .
```

Listing A.23: GIS Service Features - Computer browser

Price

The attribute *hasPrice* in Listing A.24 of the the *USDL-Pricing* module is used for set a price. *GoodRelations* vocabulary is used for specifying the currency, currency value and the unit of measurement.

```
1 price:hasPrice
2   [ a gr:UnitPriceSpecification ;
3     gr:hasCurrency "EUR" ;
4     gr:hasCurrencyValue "184.50" ;
5     gr:hasUnitOfMeasurement "MON" ] .
```

Listing A.24: GIS Price Component - Part 8 - Price)

Now we have to repeat this process for all the price plans we wish to define, and for the other two process components, *Price Module Itineraries 1 Month* and *Price Module Statics 1 Month*.

A.5.6 Service Vocabulary

As previously mentioned in section A.4, a service vocabulary is needed to express a service. This vocabulary has a number of classes, each one describing a concept of the service. A concept can be a resource, feature or other relevant information about the service.

On this section it will be explained all the concepts used, so that can be possible to reduce his subjectivity. All the information was directly retrieved

APPENDIX A. GIS USE CASE

from the following website: ⁹

```
1 :General_Features a rdfs:Class , skos:Concept ;
2   rdfs:subClassOf gr:QualitativeValue ;
3   skos:prefLabel "General Features"@en ;
4   skos:narrower
5     :Operations_POI ,
6     :Category_lists ,
7     :Coordinates ,
8     :Map_views ,
9     :Itineraries ,
10    :Data_import ,
11    :Statics ,
12    :Manage_entities .
```

Listing A.25: General Features Concept

By using the attribute *narrower* in Listing A.25 of the *SKOS* ontology we are specifying that all the below concepts are children of the general features concept. In the opposite way in Listing A.26 we have the child using the *broader* attribute meaning that the below class is the father of the current concept, this meant that the *POI_operations* concept is one of the *General_Features* concepts.

```
1 :Operations_POI a rdfs:Class , skos:Concept ;
2   rdfs:subClassOf gr:QualitativeValue ;
3   skos:prefLabel "Allows you to search, create and edit points of interest"@en
4   ;
5   skos:broader :General_Features .
```

Listing A.26: POI Operations

A.5.7 General Features

General features are the basic concepts of the service. This are the first concepts that the consumer checks before buying the Service.

POI Operations Allows the clients to search, create and edit points of interest and it's details: address, telephone, e-mail, website, foto, and so on.

Category Lists The information contained in GIS Studio is organized by categories and subcategories in accordance to the different areas.

Coordinates When activated, this option allows you to view the coordinates when moving the mouse over the map.

⁹<https://store.services.sapo.pt/en/Catalog/maps/gis-studio>

A.5. LINKED-USDL SERVICE MODELING

Map Views GIS Studios enables the user to choose from four different Map views, as such, Map, Satelite, Hybrid and Ground.

Itineraries Allows the clients to adding, edit, and delete itineraries as well as availability of these on your site; information.

Data Import Importation of files in kml or shapefile format, conversion of several or different coordinate systems allowing a greater compatibility between the various existing formats;

Statics GIS module which shows the statistical indicators about quantitative data on various economic and social sectors;

Manage Entities Allows the management of GIS information related to user permissions of your entity.

A.5.8 Advantages

This are the GIS Studio advantages. On par with the General Features, this one is also very important, so that the user can know why should buy this service.

Collaboration Between Teams GIS Studio allows the different departments of your organization collaborate simultaneously on the data management;

Easy to Use It is easy to use, featuring a design professional and intuitive. It requires no expertise in geographic information systems (GIS) for the most common tasks.

Without Extra Expensives Works in any browser or operating system and doesn't require installation of any software on the client or server.

A.5.9 Security Features

The data security it's one of the top priorities. The security features were found through out the website, and they are the same for the different price plans.

Sending and Receiving Data The process of sending and receiving data between the application and the service is done through HTTPS protocol.

APPENDIX A. GIS USE CASE

Encrypted Data The data is encrypted and signed to guarantee its integrity and confidentiality.

User Permissions Access to GIS Studio application is controlled through login and permissions assigned to each user.

A.5.10 Customer Support

GIS Studio Service has tree different ways of customer support. A good Service support is halfway to success. Since this concepts are self-explanatory there is no need for a full explanation:

1. SOS Support;
2. Mail Support;
3. Blog Maps.

A.5.11 Service Requirements

As the Service is a SaaS, the requirements are few. The user must have a computer with a browser and Internet access, and will have to authenticate to access the application.

1. Computer and Browser;
2. SAPO Mail Account.

A.5.12 Overview of GIS Use Case

This Use Case allowed us to learn more about modelling a Linked-USDL service, which in this case it was possible to see what kind of information was available in the website in order to complete the SDB data scheme. As we can see in Table A.1 thought the website we added the price component and some information about security and legal aspects. It was also added general features and advantages when we modelled the Use Case, so if we compared to SDB data scheme it was given a rather large step in the description of the service.

Table A.1: GIS Use Case and SDB data schema differences

A.5. LINKED-USDL SERVICE MODELING

		GIS Use Case	SDB Data Schema
Service Level	Performance	×	×
	Dependability	×	×
	Security	✓	Partial
	Rating	×	×
Marketing	Price	✓	×
	Documentation	✓	✓
	Certifications	×	×
Legal	Legal	Partial	×
Participants	Provider	✓	✓
	Consumer	×	×
	Partners	×	×
Bundling	Bundling	×	×
Technical	Invocation	✓	✓
	Execution	✓	✓
Operational	Operations	Partial	Partial
	Functional Description	✓	✓

B

Requirements Lists

This appendix lists all the requirements of this project according to the FURPS+[1] methodology and prioritized with the MoSCoW[18] method. The lists are organized into two sections: Functional Requirements in Section B.1 and Non-functional requirements in Section B.2.

B.1 Functional Requirements

Table B.1: Functional Requirements list

ID	Name	Priority	Verification
01	Choose remote files The system shall allow the user to choose remote files.	Must	Accomplished
02	Choose multiple files The system shall allow the user to choose multiple files.	Should	Accomplished
03	Restriction on file selection The system must verify if the file have a specific extension in order to show the information.	Should	Accomplished
04	File validation before export The system must verify if the files are valid before starting the export.	Must	Accomplished
05	Download the resulting file After the complete export the user have an option to download de resulting file.	Should	Accomplished
06	Possibility of making changes on the fly The system allows the user to change the loaded file and the resulting file on the fly.	Could	Accomplished
07	Files are automatically formatted After the selection or the export, files are automatically formatted.	Could	Accomplished
08	The system could allow the search for services The system could allow a search for finding some services described in Linked-USDL and a specific Taxonomy.	Won't	Not Accomplished

B.2 Non-functional Requirements

This section deals with the Non-functional requirements. This requirements deal with system properties rather than functions performed by the system.

APPENDIX B. REQUIREMENTS LISTS

B.2.1 Usability Requirement List

The Usability requirements relate to user interface aesthetics and data presentation to the user. The requirements can be seen in Table B.2.

Table B.2: Usability Requirements list

ID	Name	Priority	Verification
01	The information loaded is presented according to their language The user interface should show the information loaded according to their language.	Could	Accomplished
02	The resulting information is presented according to their language The user interface should show the resulting information according to their language.	Could	Accomplished
03	Errors and warnings The user interface must presented the user with warning messages if the data are not well formatted.	Should	Accomplished
04	Resulting search files The user interface should presented the resulting files in a structure way.	Won't	Not Accomplished

B.2.2 Reliability Requirement List

The Reliability requirements focused in system accuracy, failover mechanisms and availability. The accuracy of the system is of great importance to properly present the final result to the user.

Table B.3: Reliability Requirements list

ID	Name	Priority	Verification
01	Correct export from SDB data scheme to USDL The export must be exactly as seen in the previously studding of the mappings.	Must	Accomplished
02	Correct export from USDL to SDB data scheme The export must be exactly as seen in the previously studding of the mappings.	Could	Accomplished
03	Correct service searching If the user wants to search for a service the engine must output the results correctly.	Won't	Not Accomplished

B.2. NON-FUNCTIONAL REQUIREMENTS

B.2.3 Design Requirement List

The Design requirements give an overview of what are the design time requirements, e.g, data models or databases needed.

Table B.4: Design Requirements list

ID	Name	Priority	Verification
01	Login protection It should be defined a login module using a database so that the information will be inaccessible to outsiders.	Could	Accomplished
02	USDL model In order to implement the export we must use the USDL model.	Must	Accomplished
03	SDB data scheme It is also needed the SDB data scheme for all the export process.	Must	Not Accomplished
04	Use of a Triple Store If possible a Triple Store can be used for storing the Service set data.	Won't	Not Accomplished
05	Use of a Taxonomy For the search engine we will use Linked-USDL with a Taxonomy enrichment in a way that the services are correctly described.	Won't	Not Accomplished

B.2.4 Implementation Requirement List

The Implementation requirements define system implementation constraints like coding constraints or architectural constraints.

Table B.5: Implementation Requirements list

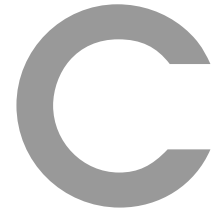
ID	Name	Priority	Verification
01	SOA architecture The architecture must implement the SOA guidelines.	Must	Accomplished
02	Programming language restriction The project must use C# language with Web services.	Must	Accomplished
03	Support libraries and programming languages In order to improve the user interface we used some additional libraries(jquery) and programming languages(asp and javascript).	Could	Accomplished

B.2.5 Interface Requirement List

The Interface requirements define constraints about external items which the system has to interact.

Table B.6: Interface Requirements list

ID	Name	Priority	Verification
01	Service set based on Linked-USDL descriptions The services are described in Linked-USDL, an RDF model. Therefore it should be possible to deal with RDF files.	Won't	Not Accomplished
02	Search engine uses Google plugin and dotNetRDF The project uses some external plugins for the search engine.	Won't	Not Accomplished



Service Lifecycle Ontology - Turtle Schema

```
1 @prefix : <http://rdfs.genssiz.org/slc#> .
2 @prefix dcterms: <http://purl.org/dc/terms/> .
3 @prefix foaf: <http://xmlns.com/foaf/0.1/> .
4 @prefix owl: <http://www.w3.org/2002/07/owl#> .
5 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
8 @prefix skos: <http://www.w3.org/2004/02/skos/core#> .
9 @prefix usdl: <http://www.linked-usdl.org/ns/usdl#> .
10
11
12 <http://rdfs.genssiz.org/SDBVocabulary>
13   rdf:type owl:Ontology ;
14   rdfs:label "Service Lifecycle" ;
15   dcterms:contributor
16     [ rdf:type foaf:Person ;
17       foaf:firstName "Filipe" ;
18       foaf:lastName "Barata" ;
19       foaf:name "Filipe Barata"
20     ] ;
21   dcterms:created "2013-05-27"^^xsd:date ;
22   dcterms:description "A vocabulary to describe the Service Lifecycle" ;
23   dcterms:modified "2013-07-08"^^xsd:date ;
24   dcterms:title "SLC Vocabulary" ;
25   owl:versionInfo "01"^^xsd:string .
26
27 <http://rdfs.genssiz.org/slc#fbarata> a foaf:Person;
28   foaf:name "Filipe Barata".
29
30 ##### CLASSES
31 #####
32 :ServiceLifecycle a rdfs:Class, owl:Class ;
33   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
34   rdfs:label "Service Lifecycle"@en ;
35   rdfs:comment "A Service Lifecycle consists in a series of phases with their
36     respective transitions between them."@en ;
37   rdfs:subClassOf usdl:Service .
```

APPENDIX C. SERVICE LIFECYCLE ONTOLOGY - TURTLE SCHEMA

```
38 :Activity a rdfs:Class, owl:Class ;
39   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
40   rdfs:label "Activity"@en ;
41   rdfs:comment "An Activity describes an instance of a Task."@en ;
42   rdfs:subClassOf :Task .
43
44 :Task a rdfs:Class, owl:Class ;
45   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
46   rdfs:label "Task"@en ;
47   rdfs:comment "A Task represents an item that must be completed in a specific
48     time."@en ;
49   rdfs:subClassOf :TaskGroup .
50
51 :Phase a rdfs:Class, owl:Class ;
52   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
53   rdfs:label "Phase"@en ;
54   rdfs:comment "A Phase defines one possible phase that a Service may be at a
55     specific Time."@en ;
56   rdfs:subClassOf :ServiceLifecycle .
57
58 :Transition a rdfs:Class, owl:Class ;
59   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
60   rdfs:label "Transition"@en ;
61   rdfs:comment "A Transition describes how a transition is made between Phases.
62     It also may describe what tasks are needed to make this transition."@en ;
63   rdfs:subClassOf :Phase .
64
65 :Role a rdfs:Class, owl:Class ;
66   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
67   rdfs:label "Role"@en ;
68   rdfs:comment "A Role defines a role or a set of roles involved in a Service
69     Phase, TaskGroup, or Task."@en ;
70   rdfs:subClassOf skos:Concept .
71
72 :TaskGroup a rdfs:Class, owl:Class ;
73   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
74   rdfs:label "Task Group"@en ;
75   rdfs:comment "A Task Group describes a set of tasks grouped together."@en ;
76   rdfs:subClassOf :Transition .
77
78 :Time a rdfs:Class, owl:Class ;
79   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
80   rdfs:label "Time"@en ;
81   rdfs:comment "Time is useful to describe the time constraints of a Phase,
82     Task or Activity. "@en .
83
84 ##### PROPERTIES
85 #####
86
87 :actualPhase a rdf:Property, owl:DatatypeProperty ;
88   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
89   rdfs:label "Actual Phase"@en ;
90   rdfs:comment "An Actual Phase serves to describe the actual phase of a
91     Service. It is important to note that a Service can only have one phase
92     at a specific time."@en ;
93   rdfs:domain :ServiceLifecycle;
94   rdfs:range :Phase .
95
96 :possiblePhase a rdf:Property, owl:DatatypeProperty ;
```

```

90 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
91 rdfs:label "Possible Phase"@en ;
92 rdfs:comment "A Possible Phase describes the different phases that are
    available in a Service Lifecycle."@en ;
93 rdfs:domain :ServiceLifecycle;
94 rdfs:range :Phase .
95
96 :resultingPhase a rdf:Property, owl:DatatypeProperty ;
97 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
98 rdfs:label "Resulting Phase"@en ;
99 rdfs:comment "A Resulting Phase is useful to describe the resulting phase
    after a successful Transition."@en ;
100 rdfs:domain [a owl:Class; owl:unionOf (:Transition :ServiceLifecycle)];
101 rdfs:range :Phase .
102
103 :activityStatus a rdf:Property, owl:DatatypeProperty ;
104 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
105 rdfs:label "Activity Status"@en ;
106 rdfs:comment "This property gives us information about the status of an
    Activity."@en ;
107 rdfs:domain :Activity;
108 rdfs:range xsd:string .
109
110 :isActivityCompleted a rdf:Property, owl:DatatypeProperty ;
111 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
112 rdfs:label "Activity Completed"@en ;
113 rdfs:comment "This property specify if the Activity is completed."@en ;
114 rdfs:domain :Activity;
115 rdfs:range xsd:boolean .
116
117 :resource a rdf:Property, owl:DatatypeProperty ;
118 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
119 rdfs:label "Resource"@en ;
120 rdfs:comment "A resource is something that is processed in the Activity."@en
    ;
121 rdfs:domain :Activity;
122 rdfs:range rdfs:Resource .
123
124 :hasTaskGroup a rdf:Property, owl:DatatypeProperty ;
125 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
126 rdfs:label "Has Task Group"@en ;
127 rdfs:comment "With this property it is possible to describe a set of Tasks."
    @en ;
128 rdfs:domain :ServiceLifecycle;
129 rdfs:range :TaskGroup .
130
131 :taskStatus a rdf:Property, owl:DatatypeProperty ;
132 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
133 rdfs:label "Task Status"@en ;
134 rdfs:comment "This property gives us information about the status of a Task."
    @en ;
135 rdfs:domain :Activity;
136 rdfs:range xsd:string .
137
138 :hasTask a rdf:Property, owl:DatatypeProperty ;
139 rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
140 rdfs:label "Has Task"@en ;
141 rdfs:comment "With this property it is possible to describe an assignment."
    @en ;
142 rdfs:domain :TaskGroup;
143 rdfs:range :Task .

```

APPENDIX C. SERVICE LIFECYCLE ONTOLOGY - TURTLE SCHEMA

```
144
145 :isTaskCompleted a rdf:Property, owl:DatatypeProperty ;
146   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
147   rdfs:label "Is Task Completed"@en ;
148   rdfs:comment "This property specifies if the Task is completed."@en ;
149   rdfs:domain :Task;
150   rdfs:range xsd:boolean .
151
152 :isTaskStarted a rdf:Property, owl:DatatypeProperty ;
153   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
154   rdfs:label "Is Task Started"@en ;
155   rdfs:comment "This property specifies if the Task has started."@en ;
156   rdfs:domain :Task;
157   rdfs:range xsd:boolean .
158
159 :isTaskStoped a rdf:Property, owl:DatatypeProperty ;
160   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
161   rdfs:label "Is Task Stoped"@en ;
162   rdfs:comment "This property specifies if the Task is on hold."@en ;
163   rdfs:domain :Task;
164   rdfs:range xsd:boolean .
165
166 :possibleTransition a rdf:Property, owl:DatatypeProperty ;
167   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
168   rdfs:label "Possible Transition"@en ;
169   rdfs:comment "A Possible Transition describes all the phases that are valid
170     or available for the next Transition."@en ;
170   rdfs:domain :Phase;
171   rdfs:range :Transition .
172
173 :startTime a rdf:Property, owl:DatatypeProperty ;
174   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
175   rdfs:label "Start Time"@en ;
176   rdfs:comment "This property gives the initial time of a specic Phase, Task or
177     Activity."@en ;
177   rdfs:domain [a :Time; owl:unionOf (:Phase :Task :Activity)];
178   rdfs:range xsd:time.
179
180 :endTime a rdf:Property, owl:DatatypeProperty ;
181   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
182   rdfs:label "End Time"@en ;
183   rdfs:comment "This property gives the ending time of a specic Phase, Task or
184     Activity."@en ;
184   rdfs:domain [a :Time; owl:unionOf (:Phase :Task :Activity)];
185   rdfs:range xsd:time.
186
187 :durationTime a rdf:Property, owl:DatatypeProperty ;
188   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
189   rdfs:label "Duration Time"@en ;
190   rdfs:comment "This property gives the duration time of a specic Phase, Task
191     or Activity"@en ;
191   rdfs:domain [a :Time; owl:unionOf (:Phase :Task :Activity)];
192   rdfs:range xsd:time.
193
194 :hasRole a rdf:Property, owl:DatatypeProperty ;
195   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
196   rdfs:label "Duration Time"@en ;
197   rdfs:comment "This property specifies a role."@en ;
198   rdfs:domain :Role ;
199   rdfs:range xsd:string .
200
```

```
201 :hasPerson a rdf:Property, owl:DatatypeProperty ;
202   rdfs:isDefinedBy <http://rdfs.genssiz.org/slc#> ;
203   rdfs:label "Duration Time"@en ;
204   rdfs:comment "This property specifies a person that belongs to a role."@en ;
205   rdfs:domain :Role ;
206   rdfs:range foaf:Person .
```

Listing C.1: Service Lifecycle Ontology

D

SAPO Services Taxonomy - Turtle Schema

This appendix describes the full list of concepts presented in the SAPO Services Taxonomy.

D.1 Top Level Concepts

There are two top level concepts in this Taxonomy:

- ***Service Model.*** Describes the service models which in our case: SaaS;
- ***Property.*** A Service property which wraps all the SAPO services properties.

D.2 Property

This class groups all the service characteristics or features. As seen in Listing D.1 there are four different categories of properties:

- ***Functional Property.*** This groups all the functional properties;
- ***Non-Functional Property.*** This groups all the non-functional properties such as integrity and confidentiality;
- ***Interface.*** Groups all the interface types or points of interaction between actors and SAPO Services;
- ***Support Property.*** This groups all the SAPO Services support properties or types of support available in the service.

APPENDIX D. SAPO SERVICES TAXONOMY - TURTLE SCHEMA

D.2.1 Functional Property

A functional property is a particular resource or feature provided by the service. With these properties we can define exactly what the service offers such as exhibit their features or advantages.

We set some categories inside the functional property in order to group resources with related characteristics. As example, we have Network Functional Property, Platform Functional Property and others.

D.2.2 Non-Functional Property

This class groups all service non-functional properties. These characteristics are usually used to judge how the service works rather than specify how it behaves. As example we have Integrity and Confidentiality described in some SAPO services.

D.2.3 Interface

This class is used to describe any type of interaction point between the service and the consumer. We use this to describe the API's and the Client interface.

D.2.4 Support Properties

Many of SAPO Services have some kind of support given by their helpdesk teams. Is an essential aspect to attract and maintain the customers.

```
1 # baseURI: http://rdfs.genssiz.org/SapoServicesTaxonomy
2
3 @prefix : <http://rdfs.genssiz.org/SapoServicesTaxonomy#> .
4 @prefix dcterms: <http://purl.org/dc/terms/> .
5 @prefix foaf: <http://xmlns.com/foaf/0.1/> .
6 @prefix gn: <http://www.geonames.org/ontology#> .
7 @prefix owl: <http://www.w3.org/2002/07/owl#> .
8 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
9 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
10 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
11 @prefix usdl: <http://www.linked-usdl.org/ns/usdl#> .
12
13
14 <http://rdfs.genssiz.org/SapoServicesTaxonomy>
15   rdf:type owl:Ontology ;
16   rdfs:label "SDB Taxonomy" ;
17   dcterms:contributor
18     [ rdf:type foaf:Person ;
19       foaf:firstName "Filipe" ;
20       foaf:lastName "Barata" ;
21       foaf:name "Filipe Barata"
22     ] ;
23   dcterms:created "2013-07-20"^^xsd:date ;
```

D.2. PROPERTY

```
24   dcterms:description "SDB properties taxonomy to be used in SUnSET project"
25   ;
26   dcterms:modified "2013-08-02"^^xsd:date ;
27   dcterms:title "SDB Properties Taxonomy" ;
28   owl:versionInfo "01"^^xsd:string .
29 ##### General #####
30
31 :Communication
32   rdf:type rdfs:Class, owl:Class ;
33   rdfs:comment "For describing any communication service."^^xsd:string ;
34   rdfs:label "Communication"^^xsd:string ;
35   rdfs:subClassOf rdfs:Class .
36
37 :Contents
38   rdf:type rdfs:Class, owl:Class ;
39   rdfs:comment "For describing any Content services."^^xsd:string ;
40   rdfs:label "Contents"^^xsd:string ;
41   rdfs:subClassOf rdfs:Class .
42
43 :Maps
44   rdf:type rdfs:Class, owl:Class ;
45   rdfs:comment "For describing any Maps service."^^xsd:string ;
46   rdfs:label "Maps"^^xsd:string ;
47   rdfs:subClassOf rdfs:Class .
48
49 :Search
50   rdf:type rdfs:Class, owl:Class ;
51   rdfs:comment "For describing any Search services."^^xsd:string ;
52   rdfs:label "Search"^^xsd:string ;
53   rdfs:subClassOf rdfs:Class .
54
55 :Development
56   rdf:type rdfs:Class, owl:Class ;
57   rdfs:comment "For describing any Development services."^^xsd:string ;
58   rdfs:label "Development"^^xsd:string ;
59   rdfs:subClassOf rdfs:Class .
60
61 :Social
62   rdf:type rdfs:Class, owl:Class ;
63   rdfs:comment "For describing any Social services."^^xsd:string ;
64   rdfs:label "Social"^^xsd:string ;
65   rdfs:subClassOf rdfs:Class .
66
67 :Advertising
68   rdf:type rdfs:Class, owl:Class ;
69   rdfs:comment "For describing any Advirtising services."^^xsd:string ;
70   rdfs:label "Advirtising"^^xsd:string ;
71   rdfs:subClassOf rdfs:Class .
72
73 :Utilities
74   rdf:type rdfs:Class, owl:Class ;
75   rdfs:comment "For describing any Utilities services."^^xsd:string ;
76   rdfs:label "Utilities"^^xsd:string ;
77   rdfs:subClassOf rdfs:Class .
78
79 :Internet
80   rdf:type rdfs:Class, owl:Class ;
81   rdfs:comment ""^^xsd:string ;
82   rdfs:label "Internet"^^xsd:string ;
83   rdfs:subClassOf rdfs:Class .
```

APPENDIX D. SAPO SERVICES TAXONOMY - TURTLE SCHEMA

```
84
85 :Messages
86   rdf:type rdfs:Class, owl:Class ;
87   rdfs:comment "Groups all the message related characteristics. These
      messages can be of a certain type (Eg: E-mail, SMS, etc) and use a
      certain protocol (HTTP, SOAP, REST, SMTP, etc).""^xsd:string ;
88   rdfs:label "Messages""^xsd:string ;
89   rdfs:subClassOf :PlatformFunctionalProperty .
90
91 :MessageType
92   rdf:type rdfs:Class, owl:Class ;
93   rdfs:label "Message type""^xsd:string ;
94   rdfs:subClassOf :Messages .
95
96 :MessageProtocol
97   rdf:type rdfs:Class, owl:Class ;
98   rdfs:label "Message protocol""^xsd:string ;
99   rdfs:subClassOf :Messages .
100
101 :MessageNumber
102   rdf:type rdfs:Class, owl:Class ;
103   rdfs:label "Message number""^xsd:string ;
104   rdfs:subClassOf :Messages .
105
106 :MessageLenght
107   rdf:type rdfs:Class, owl:Class ;
108   rdfs:label "Message lenght""^xsd:string ;
109   rdfs:subClassOf :Messages .
110
111 :MessageSender
112   rdf:type rdfs:Class, owl:Class ;
113   rdfs:label "Message sender""^xsd:string ;
114   rdfs:subClassOf :Messages .
115
116 :MessageStatus
117   rdf:type rdfs:Class, owl:Class ;
118   rdfs:label "Message status""^xsd:string ;
119   rdfs:subClassOf :Messages .
120
121 :MessageNotifications
122   rdf:type rdfs:Class, owl:Class ;
123   rdfs:label "Message notifications""^xsd:string ;
124   rdfs:subClassOf :Messages .
125
126 :MessageRecipientType
127   rdf:type rdfs:Class, owl:Class ;
128   rdfs:comment "The message type could be a single recipient or a list of
      contacts."^xsd:string ;
129   rdfs:label "Message recipient type."^xsd:string ;
130   rdfs:subClassOf :Messages .
131
132 :Subscribe
133   rdf:type rdfs:Class, owl:Class ;
134   rdfs:comment "Subscription of a specific service."^xsd:string ;
135   rdfs:label "Subscribe""^xsd:string ;
136   rdfs:subClassOf :FunctionalProperty .
137
138 :Security
139   rdf:type rdfs:Class, owl:Class ;
140   rdfs:comment "Security policies and/or protocols used."^xsd:string ;
141   rdfs:label "Security""^xsd:string ;
```

```
142     rdfs:subClassOf :NonFunctionalProperty .
143
144 :NonFunctionalProperty
145     rdf:type rdfs:Class, owl:Class ;
146     rdfs:comment "Groups all the non-functional service properties"^^xsd:
        string ;
147     rdfs:label "Non functional property"^^xsd:string ;
148     rdfs:subClassOf :Property .
149
150 :Authentication
151     rdf:type rdfs:Class, owl:Class ;
152     rdfs:label "Authentication"^^xsd:string ;
153     rdfs:subClassOf :Security .
154
155 :Credentials
156     rdf:type rdfs:Class, owl:Class ;
157     rdfs:label "Credentials"^^xsd:string ;
158     rdfs:subClassOf :Authentication.
159
160 :Market
161     rdf:type rdfs:Class, owl:Class ;
162     rdfs:label "Market"^^xsd:string ;
163     rdfs:subClassOf rdfs:Class.
164
165 :BusinessSupport
166     rdf:type rdfs:Class, owl:Class ;
167     rdfs:label "Business support"^^xsd:string ;
168     rdfs:subClassOf :SupportProperties.
169
170 :SupportProperties
171     rdf:type rdfs:Class, owl:Class ;
172     rdfs:comment "Groups all the cloud service support properties or types of
        support available in the service."^^xsd:string ;
173     rdfs:label "Support properties"^^xsd:string ;
174     rdfs:subClassOf :Property .
175
176 :API rdf:type rdfs:Class, owl:Class ;
177     rdfs:comment "For API like interfaces."^^xsd:string ;
178     rdfs:label "API"^^xsd:string ;
179     rdfs:subClassOf :Interface .
180
181 :APICalls
182     rdf:type rdfs:Class, owl:Class ;
183     rdfs:comment "For the amount of API calls or accesses to the service."^^
        xsd:string ;
184     rdfs:label "APICalls"^^xsd:string ;
185     rdfs:subClassOf :PlatformFunctionalProperty .
186
187 :PlatformFunctionalProperty
188     rdf:type rdfs:Class, owl:Class ;
189     rdfs:comment "Groups all the Platform as a Service and Software as a
        Service related characteristics."^^xsd:string ;
190     rdfs:label "Platform functional property"^^xsd:string ;
191     rdfs:subClassOf :FunctionalProperty .
192
193 :FunctionalProperty
194     rdf:type rdfs:Class, owl:Class ;
195     rdfs:comment "Groups all the service functional properties."^^xsd:string ;
196     rdfs:label "Functional property"^^xsd:string ;
197     rdfs:subClassOf :Property .
198
```

APPENDIX D. SAPO SERVICES TAXONOMY - TURTLE SCHEMA

```
199 :Interface
200     rdf:type rdfs:Class, owl:Class ;
201     rdfs:comment "Groups all the types of interfaces or points of interaction
                between actors and the Service."^^xsd:string ;
202     rdfs:label "Interface"^^xsd:string ;
203     rdfs:subClassOf :Property .
204
205 :Property
206     rdf:type rdfs:Class, owl:Class ;
207     rdfs:comment "A service property, wraps all the service properties."^^xsd:
                string ;
208     rdfs:label "Property"^^xsd:string ;
209     rdfs:subClassOf rdfs:Class .
210
211 :RESTAPI
212     rdf:type rdfs:Class, owl:Class ;
213     rdfs:label "REST API"^^xsd:string ;
214     rdfs:subClassOf :APICalls .
215
216 :SOAPAPI
217     rdf:type rdfs:Class, owl:Class ;
218     rdfs:label "SOAP API"^^xsd:string ;
219     rdfs:subClassOf :APICalls .
220
221 :NetworkFunctionalProperty
222     rdf:type rdfs:Class, owl:Class ;
223     rdfs:comment "Groups all the network related characteristics."^^xsd:string
                ;
224     rdfs:label "Network functional property"^^xsd:string ;
225     rdfs:subClassOf :FunctionalProperty .
226
227 :WebPages
228     rdf:type rdfs:Class, owl:Class ;
229     rdfs:comment "For services that deal with web hosting or any website
                activity."^^xsd:string ;
230     rdfs:label "Web Pages"^^xsd:string ;
231     rdfs:subClassOf :PlatformFunctionalProperty .
232
233 :Applications
234     rdf:type rdfs:Class, owl:Class ;
235     rdfs:comment "For describing any constraints or features about the amount
                of application allowed."^^xsd:string ;
236     rdfs:label "Applications"^^xsd:string ;
237     rdfs:subClassOf :PlatformFunctionalProperty .
238
239 :DesktopApplications
240     rdf:type rdfs:Class, owl:Class ;
241     rdfs:label "Desktop Applications"^^xsd:string ;
242     rdfs:subClassOf :Applications .
243
244 :Management
245     rdf:type rdfs:Class, owl:Class ;
246     rdfs:comment "For describing any management properties, as example,
                service management."^^xsd:string ;
247     rdfs:label "Management"^^xsd:string ;
248     rdfs:subClassOf :FunctionalProperty .
249
250 :Contacts
251     rdf:type rdfs:Class, owl:Class ;
252     rdfs:label "Contacts"^^xsd:string ;
253     rdfs:subClassOf :Management .
```



```
254
255 :Webmail
256     rdf:type rdfs:Class, owl:Class ;
257     rdfs:label "Webmail"^^xsd:string ;
258     rdfs:subClassOf :PlatformFunctionalProperty .
259
260 :Events
261     rdf:type rdfs:Class, owl:Class ;
262     rdfs:label "Events"^^xsd:string ;
263     rdfs:subClassOf :Agenda .
264
265 :Areas
266     rdf:type rdfs:Class, owl:Class ;
267     rdfs:comment "For describing different type of Areas like Geographical
268     Areas."^^xsd:string ;
269     rdfs:label "Areas"^^xsd:string ;
270     rdfs:subClassOf :Parameters.
271
272 :Parameters
273     rdf:type rdfs:Class, owl:Class ;
274     rdfs:comment "For describing different Parameters such as geographical
275     coordinates, a location, phone number, etc."^^xsd:string ;
276     rdfs:label "Parameters"^^xsd:string ;
277     rdfs:subClassOf :FunctionalProperty .
278
279 :Categories
280     rdf:type rdfs:Class, owl:Class ;
281     rdfs:comment "For describing different Categories such as Hotels,
282     Restaurants, etc."^^xsd:string ;
283     rdfs:label "Categories"^^xsd:string ;
284     rdfs:subClassOf :Parameters .
285
286 :DataFunctionalProperty
287     rdf:type rdfs:Class, owl:Class ;
288     rdfs:label "Data functional property"^^xsd:string ;
289     rdfs:subClassOf :FunctionalProperty .
290
291 :DataRequest
292     rdf:type rdfs:Class, owl:Class ;
293     rdfs:comment "groups all the database related requests"^^xsd:string ;
294     rdfs:label "Request"^^xsd:string ;
295     rdfs:subClassOf :DataFunctionalProperty .
296
297 :Database
298     rdf:type rdfs:Class, owl:Class ;
299     rdfs:label "Applications"^^xsd:string ;
300     rdfs:subClassOf :DataFunctionalProperty .
301
302 :SupportProperties
303     rdf:type rdfs:Class, owl:Class ;
304     rdfs:comment "Groups all the service support properties or types of
305     support available in the service."^^xsd:string ;
306     rdfs:label "Support properties"^^xsd:string ;
307     rdfs:subClassOf :Property .
308
309 :SupportTeam
310     rdf:type rdfs:Class, owl:Class ;
311     rdfs:comment "for services that provide a specialized support team to its
312     users."^^xsd:string ;
313     rdfs:label "Support team"^^xsd:string ;
314     rdfs:subClassOf :SupportProperties .
```

APPENDIX D. SAPO SERVICES TAXONOMY - TURTLE SCHEMA

```
310
311 :SpecialistTeams
312     rdf:type rdfs:Class, owl:Class ;
313     rdfs:label "Specialist of Teams"^^xsd:string ;
314     rdfs:subClassOf :SupportTeam .
315
316 :GeoreferencedContent
317     rdf:type rdfs:Class, owl:Class ;
318     rdfs:label "Georeferenced Content"^^xsd:string ;
319     rdfs:subClassOf :POI .
320
321 :GeoMarketing
322     rdf:type rdfs:Class, owl:Class ;
323     rdfs:comment "Support on decision-making in GeoMarketing studies."^^xsd:
324         string ;
325     rdfs:label "GeoMarketing"^^xsd:string ;
326     rdfs:subClassOf :SupportProperties .
327
328 :Requests
329     rdf:type rdfs:Class, owl:Class ;
330     rdfs:comment "for describing any type of data Request."^^xsd:string ;
331     rdfs:label "Requests"^^xsd:string ;
332     rdfs:subClassOf :DataRequest .
333
334 :Sources
335     rdf:type rdfs:Class, owl:Class ;
336     rdfs:comment "for specifying any kind of Source."^^xsd:string ;
337     rdfs:label "Sources"^^xsd:string ;
338     rdfs:subClassOf :Parameters .
339
340 :IPTV
341     rdf:type rdfs:Class, owl:Class ;
342     rdfs:label "IPTV"^^xsd:string ;
343     rdfs:subClassOf :PlatformFunctionalProperty, :MEOEPG .
344
345 :GameOver
346     rdf:type rdfs:Class, owl:Class ;
347     rdfs:label "GameOver"^^xsd:string ;
348     rdfs:subClassOf :GamesRSS .
349
350 :RSS
351     rdf:type rdfs:Class, owl:Class ;
352     rdfs:label "RSS"^^xsd:string ;
353     rdfs:subClassOf :DataFunctionalProperty, :GamesRSS .
354
355 :Semantic
356     rdf:type rdfs:Class, owl:Class ;
357     rdfs:comment "for describing any service with semantic properties."^^xsd:
358         string ;
359     rdfs:label "Semantic"^^xsd:string ;
360     rdfs:subClassOf :FunctionalProperty, :SemanticLists .
361
362 :Words
363     rdf:type rdfs:Class, owl:Class ;
364     rdfs:label "Words"^^xsd:string ;
365     rdfs:subClassOf :Semantic .
366
367 :WebApplications
368     rdf:type rdfs:Class, owl:Class ;
369     rdfs:label "Web Applications"^^xsd:string ;
370     rdfs:subClassOf :Applications .
```

D.2. PROPERTY

```
369
370 :SaaS
371     rdf:type rdfs:Class, owl:Class ;
372     rdfs:comment "See description by NIST in http://csrc.nist.gov/publications
373         /nistpubs/800-145/SP800-145.pdf"^^xsd:string ;
374     rdfs:label "Software as a Service"^^xsd:string ;
375     rdfs:subClassOf :ServiceModel .
376 :ServiceModel
377     rdf:type rdfs:Class, owl:Class ;
378     rdfs:label "Service model"^^xsd:string ;
379     rdfs:subClassOf rdfs:Class .
380
381 :Browser
382     rdf:type rdfs:Class, owl:Class ;
383     rdfs:label "Browser"^^xsd:string ;
384     rdfs:subClassOf :DesktopApplications .
385
386 :OperatingSystem
387     rdf:type rdfs:Class, owl:Class ;
388     rdfs:comment "Groups all the Operating system related characteristics.
389         They can be of the types: Embedded, Mobile, Real Time, Unix or Windows
390         ."^^xsd:string ;
391     rdfs:label "Operating system"^^xsd:string ;
392     rdfs:subClassOf :ComputingFunctionalProperty .
393
394 :ComputingFunctionalProperty
395     rdf:type rdfs:Class, owl:Class ;
396     rdfs:comment "Groups all the computational (IaaS) resources related
397         characteristics."^^xsd:string ;
398     rdfs:label "Computing functional property"^^xsd:string ;
399     rdfs:subClassOf :FunctionalProperty .
400
401 :Software
402     rdf:type rdfs:Class, owl:Class ;
403     rdfs:label "Software"^^xsd:string ;
404     rdfs:subClassOf :PlatformFunctionalProperty .
405
406 :Client
407     rdf:type rdfs:Class, owl:Class ;
408     rdfs:label "Client"^^xsd:string ;
409     rdfs:subClassOf :Interface.
410
411 :Server
412     rdf:type rdfs:Class, owl:Class ;
413     rdfs:label "Server"^^xsd:string ;
414     rdfs:subClassOf :ComputingFunctionalProperty .
415
416 :PointsOfInterest
417     rdf:type rdfs:Class, owl:Class ;
418     rdfs:label "Points of Interest"^^xsd:string ;
419     rdfs:subClassOf :Parameters, :POI .
420
421 :Itineraries
422     rdf:type rdfs:Class, owl:Class ;
423     rdfs:label "Itineraries"^^xsd:string ;
424     rdfs:subClassOf :Parameters, :POI .
425
426 :Supervision
427     rdf:type rdfs:Class, owl:Class ;
428     rdfs:label "Supervision"^^xsd:string ;
```

APPENDIX D. SAPO SERVICES TAXONOMY - TURTLE SCHEMA

```
426     rdfs:subClassOf :Management, :FunctionalProperty .
427
428 :DataImport
429     rdf:type rdfs:Class, owl:Class ;
430     rdfs:label "Data Import"^^xsd:string ;
431     rdfs:subClassOf :Management, :DataFunctionalProperty .
432
433 :Statics
434     rdf:type rdfs:Class, owl:Class ;
435     rdfs:label "Statics"^^xsd:string ;
436     rdfs:subClassOf :DataFunctionalProperty .
437
438 :Entities
439     rdf:type rdfs:Class, owl:Class ;
440     rdfs:comment "Management of information related to user permissions of the
441         entity."^^xsd:string ;
442     rdfs:label "Entities"^^xsd:string ;
443     rdfs:subClassOf :Management, :DataFunctionalProperty .
444
445 :HTTPS
446     rdf:type rdfs:Class, owl:Class ;
447     rdfs:label "HTTPS"^^xsd:string ;
448     rdfs:subClassOf :NetworkFunctionalProperty .
449
450 :Encrypted
451     rdf:type rdfs:Class, owl:Class ;
452     rdfs:label "Encrypted"^^xsd:string ;
453     rdfs:subClassOf :Security, :Integrity .
454
455 :Signed
456     rdf:type rdfs:Class, owl:Class ;
457     rdfs:label "Signed"^^xsd:string ;
458     rdfs:subClassOf :Security, :Confidentiality .
459
460 :Integrity
461     rdf:type rdfs:Class, owl:Class ;
462     rdfs:comment "for Integrity related features or characteristics."^^xsd:
463         string ;
464     rdfs:label "Integrity"^^xsd:string ;
465     rdfs:subClassOf :NonFunctionalProperty .
466
467 :Confidentiality
468     rdf:type rdfs:Class, owl:Class ;
469     rdfs:comment "for Confidentiality related features or characteristics."^^
470         xsd:string ;
471     rdfs:label "Confidentiality"^^xsd:string ;
472     rdfs:subClassOf :NonFunctionalProperty .
473
474 :Login
475     rdf:type rdfs:Class, owl:Class ;
476     rdfs:comment "For describing any type of service login."^^xsd:string ;
477     rdfs:label "Login"^^xsd:string ;
478     rdfs:subClassOf :FunctionalProperty, :Security .
479
480 :Proxy
481     rdf:type rdfs:Class, owl:Class ;
482     rdfs:comment "For describing any intermediary server in the request or
483         post process."^^xsd:string ;
484     rdfs:label "Proxy"^^xsd:string ;
485     rdfs:subClassOf :NetworkFunctionalProperty .
```

```
483 :CodeLanguages
484     rdf:type rdfs:Class, owl:Class ;
485     rdfs:comment "For specifying programming languages supported by the
         service."^^xsd:string ;
486     rdfs:label "Code Languages"^^xsd:string ;
487     rdfs:subClassOf :PlatformFunctionalProperty .
488
489 :Comments
490     rdf:type rdfs:Class, owl:Class ;
491     rdfs:label "Comments"^^xsd:string ;
492     rdfs:subClassOf :FunctionalProperty .
493
494 :Drafts
495     rdf:type rdfs:Class, owl:Class ;
496     rdfs:label "Drafts"^^xsd:string ;
497     rdfs:subClassOf :FunctionalProperty .
498
499 :Posts
500     rdf:type rdfs:Class, owl:Class ;
501     rdfs:label "Posts"^^xsd:string ;
502     rdfs:subClassOf :FunctionalProperty .
503
504 :Limits
505     rdf:type rdfs:Class, owl:Class ;
506     rdfs:comment "For describing the hosting limits, such as, the number of free
         photos."^^xsd:string ;
507     rdfs:label "Limits"^^xsd:string ;
508     rdfs:subClassOf :FunctionalProperty .
509
510 :Duration
511     rdf:type rdfs:Class, owl:Class ;
512     rdfs:comment "For describing the duration limits, such as, the maximum
         duration of a video."^^xsd:string ;
513     rdfs:label "Duration"^^xsd:string ;
514     rdfs:subClassOf :FunctionalProperty .
515
516 :Share
517     rdf:type rdfs:Class, owl:Class ;
518     rdfs:comment "For describing any sharing properties."^^xsd:string ;
519     rdfs:label "Share"^^xsd:string ;
520     rdfs:subClassOf :FunctionalProperty .
521
522 :Upload
523     rdf:type rdfs:Class, owl:Class ;
524     rdfs:comment "For describing any Upload properties."^^xsd:string ;
525     rdfs:label "Upload"^^xsd:string ;
526     rdfs:subClassOf :FunctionalProperty .
527
528 :Watch
529     rdf:type rdfs:Class, owl:Class ;
530     rdfs:label "Watch"^^xsd:string ;
531     rdfs:subClassOf :FunctionalProperty .
532
533 :Aggregator
534     rdf:type rdfs:Class, owl:Class ;
535     rdfs:comment "For describing any service aggregator property."^^xsd:string
         ;
536     rdfs:label "Aggregator"^^xsd:string ;
537     rdfs:subClassOf :FunctionalProperty .
538
539 :Publisher
```

APPENDIX D. SAPO SERVICES TAXONOMY - TURTLE SCHEMA

```
540     rdf:type rdfs:Class, owl:Class ;
541     rdfs:label "Publisher"^^xsd:string ;
542     rdfs:subClassOf :FunctionalProperty .
543
544 :FollowFriends
545     rdf:type rdfs:Class, owl:Class ;
546     rdfs:label "Follow Friends"^^xsd:string ;
547     rdfs:subClassOf :FunctionalProperty .
548
549 :Platform
550     rdf:type rdfs:Class, owl:Class ;
551     rdfs:label "Platform"^^xsd:string ;
552     rdfs:subClassOf :PlatformFunctionalProperty .
553
554 :Image
555     rdf:type rdfs:Class, owl:Class ;
556     rdfs:label "Image"^^xsd:string ;
557     rdfs:subClassOf :FunctionalProperty, :Parameters .
558
559 :URLRequest
560     rdf:type rdfs:Class, owl:Class ;
561     rdfs:comment "groups all the URL related requests"^^xsd:string ;
562     rdfs:label "URL Request"^^xsd:string ;
563     rdfs:subClassOf :DataRequest .
564
565 :CompressedURLs
566     rdf:type rdfs:Class, owl:Class ;
567     rdfs:comment ""^^xsd:string ;
568     rdfs:label "Compressed URLs"^^xsd:string ;
569     rdfs:subClassOf :DataRequest .
```

Listing D.1: SAPO Services Taxonomy