

A Semantic Contract Model and Knowledge-driven Process for Supporting Controllability in Service-oriented Approaches

Gloria Elena Jaramillo

▶ **To cite this version:**

Gloria Elena Jaramillo. A Semantic Contract Model and Knowledge-driven Process for Supporting Controllability in Service-oriented Approaches. Cryptography and Security [cs.CR]. Université de Pau et des Pays de l'Adour, 2016. English. NNT : . tel-02413034

HAL Id: tel-02413034

https://univ-pau.hal.science/tel-02413034

Submitted on 16 Dec 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Semantic Contract Model and Knowledge-driven Process for Supporting Controllability in Service-oriented Approaches

by

Gloria Elena JARAMILLO

A dissertation submitted in partial fulfillment of the requirements for the degree of

PhD in Computer Science
University of Pau and Pays de l'Adour (UPPA)
Doctoral School of Exact Sciences and their Applications

DOCTORAL DISSERTATION COMMITTEE

Camille SALINESI	Full Professor - Université Paris 1 - Panthéon Sorbonne	Committee Chair
Ernesto DAMIANI	Full Professor - Università degli Studi di Milano	Reviewer
Maria Teresa GÓMEZ	Associate Professor - Universidad de Sevilla	Reviewer
Philippe ANIORTE	Full Professor - Université de Pau et des Pays l'Adour	Dissertation Advisor
Manuel MUNIER	Associate Professor - Université de Pau et des Pays l'Adour	Dissertation Co-advisor

Un modèle sémantique de contrat et un processus basé sur la connaissance pour garantir la contrôlabilité dans les approches orientées services

par

Gloria Elena JARAMILLO

Presentée et soutenue publiquement le 12 décembre 2016
pour obtenir le grade de

Docteur - Specialité Informatique
Université de Pau et des Pays de l'Adour
École Doctorale, Sciences Exactes et leurs Applications

JURY

Camille SALINESI	Professeur des Universités - Université Paris 1 - Panthéon Sorbonne	Président du jury
Ernesto DAMIANI	Professeur des Universités - Università degli Studi di Milano	Rapporteur
Maria Teresa GÓMEZ	Professeur - Universidad de Sevilla	Rapporteur
Philippe ANIORTE	Professeur des Universités - Université de Pau et des Pays l'Adour	Directeur de Thèse
Manuel MUNIER	Maître de Conférences - Université de Pau et des Pays l'Adour	Co-encadrant de Thèse

Abstract

Service-oriented paradigms have dramatically changed the way of providing applications and doing businesses. Both SOA and the Cloud have enabled the creation of new paradigms based on dynamic collaborations. For final users, services offer a simplified access to functionalities and data. For organizations, the delegation of some business processes and the integration of external processes into the business logic have represented an opportunity to generate competitive advantages by saving costs, increasing the visibility in the market and exploiting the expertise of their partners while offering added-value product and services to their clients.

Despite the attractive advantages of the service-based technologies, the inherent loss of control on the exchanged resources is a well-reported drawback which generates reluctance, and decelerate their wide adoption. Basically, within organizations several business rules are associated to their resources in order to keep some properties over them and ensure that they are correctly used. Such rules are associated not only access control but any condition leading to prevent possible organizational damages; for instance, conditions ensuring the client satisfaction, preventing the loss of reputation or guaranteeing compliance with some standard or legal normative. However, from the moment the resource moves out beyond the walls of the organization there is no guarantee about whether the resource is used by respecting those established rules. The consequences of such loss of control on the usage are not trivial since the way in which the shared resources are used by the external partner may intentionally or unintentionally affect the organization by causing monetary fines, loss of customers or lawsuits. The impacts of such damages justify the need to having methods aiming to control the use on the shared resources during an external service provision. In this scenario, the challenge is how to guarantee that the external partner behave as expected when the resource is in the domain of the external partner, and when the business dependence of each organization needs to be preserved.

This dissertation proposes that the service provision, including the interaction between clients and providers, be governed by a service contract. This contract differs from traditional SLAs in several ways:

- It extends the expressiveness of the SLA service guarantees, traditionally based on security and performance, with contractual terms representing business requirements about the expected use of resources
- It is based on a formal semantics which avoids misinterpretations of the contractual terms thanks to a common understanding of their meaning.

- The compliance with the business requirements is inferred from the available knowledge collected during the contract execution.

The dissertation's contribution is framed within a governance approach which aims not only at the creation of policies but also to the implementation of some processes which evaluate and give feedback to those policies. Consequently, the proposed controllability method is supported by two building blocks: a modeling component and a process component. Regarding the modeling component, two complementary models are proposed. The first one is a generic semantic formalization of a service contract which includes the definition of a controllability vocabulary. The second one is a specific model for the definition of controllability policies, which formalizes the expected behavior to the contractual parties and uses the semantic model to give a clear meaning to the definition of each rule within the policy. In the process component, the available knowledge about the behavior of the contractual parties is used to verify the compliance with the policy. Similarly, such a knowledge, which has been recorded in a log, is further exploited to assess the quality of the provided service and the behavior of the external partner.

proposed method is validated through the creation of machine-readable contracts in OWL which contain controllability policies written as XML rules. Similarly, a log is implemented acting as a knowledge base where some reasoning leading the auditing of the contractual parties is proposed.

The implemented knowledge-based reasoning, opens new perspectives about the implementation of more sophisticated techniques of artificial intelligence applied to web services, improving existing research domains such as the semantic web services and intelligent web services. On the other hand, this thesis leaves some aspects untreated, notably, the negotiation of the contractual policy.

Keywords: Contract, Model, Semantic, Controllability Policy, Business Rules, Log, Behavior, Organizational Assets, Service.

Résumé

Les paradigmes orientées service ont considérablement changé la façon de construire les applications et de faire des affaires. Tant l'approche SOA que le Cloud ont permis l'émergence de nouveaux modèles basés sur des collaborations dynamiques. Du point de vue des utilisateurs finaux, les services offrent un accès simplifié aux fonctionnalités et aux données. Quant aux organisations, la délégation de certains processus métier et l'intégration des processus externes dans la logique organisationnelle représentent une opportunité de générer des avantages concurrentiels en réduisant les coûts, en augmentant la visibilité sur le marché et en exploitant l'expertise de leurs partenaires en offrant à ses clients des produits et des services avec de la valeur ajoutée.

Malgré les avantages attrayants des technologies basées sur les services, la perte de contrôle inhérente sur les ressources échangées est un inconvénient bien connu qui génère de la réticence et freine leur large adoption. Fondamentalement, au sein des organisations différent types des règles sont associés aux ressources afin de garantir certaines de leurs propriétés et veiller à ce qu'elles soient correctement utilisées. Des telles règles sont associées non seulement au contrôle d'accès, mais également á n'importe quelle condition visant à prévenir d'éventuels dommages organisationnels. Nous pouvons citer par exemple les conditions assurant la satisfaction des clients, la prévention de la perte de réputation ou la garantie de la conformité avec une norme juridique ou standard. Cependant, à partir du moment où la ressource sort du périmètre de l'organisation, il n'y a plus aucun moyen de savoir si la ressource est utilisée en respectant les règles établies. Les conséquences d'une telle perte de contrôle sur l'utilisation ne sont pas négligeables puisque la façon dont les ressources partagées sont utilisées par le partenaire externe peut de manière intentionnel ou non, affecter l'organisation en entraînant des pénalités, la perte de clients ou des poursuites. Les impacts de ces dommages justifient la nécessité d'avoir des méthodes visant à contrôler l'utilisation des ressources partagées lors d'une prestation de services externes. Dans ce scénario, le défi est de savoir comment garantir que le partenaire externe se comporte comme prévu lorsque la ressource est dans son domaine et lorsque la dépendance à l'égard des affaires de chaque organisation doit être préservé.

Cette thèse propose que la prestation de services, y compris l'interaction entre les clients et les fournisseurs, soit régie par un contrat de service. Ce contrat diffère de SLA traditionnels de plusieurs façons:

- Il augmente l'expressivité des garanties de service SLA, traditionnellement basées sur la sécurité et la performance, avec des termes contractuels représentant les exigences opérationnelles sur l'utilisation prévue des res-

sources.

- Il est basé sur une sémantique formelle qui évite des erreurs d'interprétation des clauses contractuelles grâce à une compréhension commune de leur signification.
- La conformité avec les exigences de l'entreprise en matière d'usage des ressources est déduite de la connaissance disponible recueillie au cours de l'exécution du contrat.

La contribution de ce travail s'inscrit dans une approche de gouvernance qui vise non seulement la définition de politiques, mais aussi la mise en œuvre de certains processus qui évaluent et donnent un feedback de ces politiques. Par conséquent, notre méthode de contrôlabilité est basée sur deux éléments. Le premier a trait à la modélisation des politiques, et la second consiste en un processus. En ce qui concerne la modélisation, deux modèles complémentaires sont proposés. Le premier cible la formalisation sémantique générique d'un contrat de service, ce qui inclut la définition d'un vocabulaire de contrôlabilité. Le second est un modèle spécifique pour la définition des politiques de contrôlabilité, qui formalise le comportement attendu des parties contractuelles et utilise le modèle sémantique pour donner une signification claire à la définition de chaque règle dans la politique. Au cours du processus, les connaissances disponibles sur le comportement des parties contractuelles est utilisé pour vérifier la conformité avec la politique. De même, cette connaissance, qui a été enregistrée dans un journal, est exploitée pour évaluer la qualité du service fourni et le comportement du partenaire externe.

La méthode proposée est validée par la création de contrats lisibles par la machine dans OWL qui contiennent des politiques de contrôlabilité écrites comme des règles XML. De même, un journal est mis en œuvre agissant comme une base de connaissances permettant d'expertiser les parties contractantes. Une telle méthode, et plus concrètement le raisonnement basé sur la connaissance, offre de nouvelles perspectives quant à la mise en œuvre des techniques plus sophistiquées de l'Intelligence Artificielle appliquée aux services web, ce qui constituerait une contribution à des domaines de recherche existants, tels que les services web sémantiques et les services web intelligents. D'autre part, cette thèse ouvre des travaux futurs, notamment la négociation de la politique contractuelle en considérant des contrats multipartites.

Mots-clés: modèle, contrat, sémantique, politique de contrôlabilité, règles métier, journal, comportement, actifs organisationnel, service.

Contents

1 Introduction 2
1.1 Context 2
1.2 Motivation 3
1.3 Problem Statement 4
1.4 Objectives and Contribution 6
1.5 Thesis Organization 7

I Framework 9

2 Background 11
2.1 Service Oriented Architectures (SOA) 11
2.1.1 Composed Services 13
2.1.2 Semantic Web Services 14
2.2 The Cloud 15
2.3 Service Level Agreement (SLA) 16
2.4 Governance 17
2.5 Normativity about the Use of Assets 19
2.6 Security in Service-oriented Environments 20
2.7 Controllability 23
2.8 Discussion 24
2.9 Conclusion 26

II Models for Controllability based on Contracts 29

3 Semantic-based Service Contract Model 31
3.1 Introduction 31
3.2 State of the Art 32
3.3 Semantic Contracts 36
3.4 A Formal Model of Semantic Contracts 43
3.4.1 Contract Header 44
Actor 44
Service 47
Attributes 48
3.4.2 Contract Terms 49
Asset 49
Controller and Processor 52
Business Activity 54

Damage and Infringement 55
3.4.3 Signature 57
3.5 Machine-readable Semantic Contract 61
3.6 Discussion 68
3.7 Conclusion 73

4 A Model for the Controllability Terms based on Rules 75
4.1 Introduction 75
4.2 State of the Art 77
4.3 Knowledge Representation based on Rules 81
4.4 Contract Term Model 84
4.4.1 Subject 85
4.4.2 Behavior 86
4.4.3 Modality 87
4.4.4 Context 88
4.4.5 Behavioral and Assignation Rules 88
4.4.6 Purpose 90
4.4.7 Rule Weight 91
4.4.8 Evidence 92
4.5 Machine-readable Contractual Terms 94
4.6 Examples 96
4.7 Discussion 104
4.8 Conclusion 108

III The Process of Asset Controllability 111

5 Knowledge-driven Reasoning on Controllability 112
5.1 Introduction 112
5.2 State of the Art 114
5.3 Semi-Automatic Auditing based on Logs 118
5.3.1 Methodology for the Semi-automatic Auditing 118
5.3.2 Log Creation 120
5.3.3 Contract Compliance - Primitive Approach 125
5.3.4 Contract Compliance - Refined Approach 128
5.3.5 Example - Contract Compliance 132
5.4 Knowledge Base Analysis 141
5.5 Conclusion 144

IV General Conclusion 146

6 Conclusion 147
6.1 Objectives and Contributions 148
6.2 Thesis Perspectives 150

Appendices 158

Appendix A DL Expressiveness 159
A.1 Expressiveness of DL-Family of Languages 159
A.2 Complexity of the DL-Family of Languages 160

List of Figures

2.1 Example of a SOAP Message 13
2.2 Services Composed with a Choreography Approach 14
2.3 Services Composed with a Orchestration Approach 14
2.4 Web Evolution 15
2.5 The Cloud Layers 16
2.6 WSLA Model 17
2.7 Governance 18
2.8 Examples of Federated Scenarios 22
2.9 WS-* Family 27
2.10 Controllability on Documents 28
2.11 Controllability Governance 28

3.1 Contracts vs SLA 38
3.2 Contract Structure 44
3.3 Contract Header Example 44
3.4 Multiples Roles for an Actor 46
3.5 Subset of Attributes 49
3.6 Semantic of the Contract Header Terms 50
3.7 Semantic of the Organizational Asset 52
3.8 Business Activity 54
3.9 Taxonomy of the Contract Terms 56
3.10 Semantics of Infringement 57
3.11 Semantic of the Contract Signatures 59
3.12 Semantic Contract Model 60
3.13 RDF Graph Example 61
3.14 OWL Example 63
3.15 Syntax / Semantics of OWL 2 64
3.16 Contract Ontology Metrics 65
3.17 Taxonomy of the Contract Ontology 66
3.18 Graphical Representation of a Contract Header Example 68
3.19 A Two-layer Semantic 72
3.20 Open World Assumption Example 74

4.1 Contract Rule Structure 90
4.2 Modal scale 91
4.3 Model for the Rule's Evidence 93
4.4 Rule-based Model for the Contractual Terms 93
4.5 Tagging of the Rule's Elements 98
4.6 Service Workflows and Contracts 109

5.1 Contract Management 120
5.2 Life-cycle of a contractual rule 129
5.3 Inference about non accomplished rules 131
5.4 Oil & Gas Example 133
5.5 Life-line of the Rule R2 135
5.6 Query about Trusted Rules 136
5.7 Facts in the Log Knowledge Base 137
5.8 Automatic Contract Evaluation in RuleML 138
5.9 Knowledge Base in CLIPS 139
5.10 Automatic Contract Evaluation in CLIPS 139
5.11 Automatic Contract Evaluation - Breached Contract Case 140
5.12 Automatic Contract Evaluation - Frame Change Case 141

List of Tables

3.1 Approaches for Representation of Service Requirements 37
3.2 Vocabulary of the Service Contracts 62
3.3 Contract Ontology Constructors 65
3.4 Concept Equivalence at the Top Level with Existing Ontologies . 71

4.1 Approaches for the Representation of Organizational Policies . . 82

5.1 State of the Contractual Rules 124
5.2 Contract Metrics 126

A.1 Expressiveness Power of DL 159

Listings

3.1 Machine-readable representation for a contract excerpt 67
3.2 SPARQL to List the Contractual Parties 72
4.1 XML Syntax for the Dropbox Term 98
4.2 XML Concrete Contract Rule Syntax 100
4.3 OWL Contract Rule for the Oil&Gas Example 103
4.4 Example of Rule Meta-information 104

5.1 Rule Metadata Oil&Gas Example 134

Acronyms

SOA Service Oriented Architecture

SLA Service Level Agreement

B2B Business-to-Business

XML eXtensible Markup Language

REST REpresentational State Transfer

WSDL Web Service Definition Language

UDDI Universal Description, Discovery and Integration

BPEL Business Process Execution Language

QoS Quality of Service

SLO Service Level Objectives

RDF Resource Description Framework

RDF(S) RDF Schema

OWL Web Ontology Language

DL Description Logic

FOL First Order Logic

OWA Open-World Assumption

CWA Close-World Assumption

SWRL Semantic Web Rule Language

LP Logic Programming

TTP Trusted-Third Party

Publications

Some articles have been accepted further to this dissertation. In particular, [3] was published in collaboration with the SESAR Lab as a result of an internship at the Università degli Studi di Milano.

[1] Guillaume Cabanac, Amira Derrdji, Ali Jaffal, Jonathan Louëdec, and Gloria Elena Jaramillo. Forum jeunes chercheurs at inforsid 2014. *Revue des sciences et technologies de l'information*, 20(2):119–143, 2015.

[2] Gloria Elena Jaramillo, Philippe Aniorte, and Manuel Munier. Accords de niveau de service et modèle de réputation pour le contrôle d'usage. In *Forum Jeunes Chercheurs à INFORSID 2014 (FJC'2014)*, pages 57–60, Lyon, France, May 2014.

[3] Gloria Elena Jaramillo, Marco Anisetti, and Claudio A Ardagna. A hybrid representation model for service contracts. In *2015 International Conference on Information and Communication Technology Research (ICTRC2015)*, Abu Dhabi, UAE, May 2015.

[4] Gloria Elena Jaramillo, Manuel Munier, and Philippe Aniorte. Information security in business intelligence based on cloud: A survey of key issues and the premises of a proposal. In *WOSIS 2013 - Proceedings of the 10th International Workshop on Security in Information Systems, In conjunction with ICEIS 2013, Angers, France, 4-7 July, 2013*, 2013.

[5] Gloria Elena Jaramillo, Manuel Munier, and Philippe Aniorte. Sécurité de l'Information dans les Environnements Inter-Organisationnels. 8ème Conférence sur la Sécurité des Architectures Réseaux et des Systèmes d'Information (SARSSI'2013), September 2013.

Chapter 1

Introduction

The growing needs of collaboration across organizations have generated deep changes in the underlying information technologies. Currently, service-based approaches represent a frenzy and promising solution, offered to facilitate the interoperability and interaction facing the complex enterprise challenges. In this chapter, the framework of this work is presented, with a special focus in arguing the lack of control on the resources shared during the provision of a service. Such an identified problem is used as a base line to present the objectives and the main contribution of this dissertation.

1.1 Context

The dynamics of global markets, free trade agreements and domestic policies of the countries have been changing the way of doing businesses. Organizations have perceived the need to be constantly innovating in order to create competitive advantages which allow them to effectively take part in the market. Thence internal policies within organizations have increasingly sought to focus on the core of their business goals while collaborate with external partners by delegating and outsourcing secondary activities. Nowadays, words such as interaction, collaboration, outsourcing and data exchange are becoming more frequent in business strategies. Technology has undoubtedly contributed to this changes by facilitating both communication and interaction among organizations.

From the technological point of view, the attractive advantages of "being connected" have led to changes in the architecture and design of the information systems, which increasingly need to be adaptable, modular and flexible to support different environments, and users as well as the dynamism of the global market tendencies. Service Oriented Architecture (SOA) and the Cloud have emerged as service-based approaches offering an effective solution to those challenges. On the one hand, the cloud aims the virtualization as an alternative to traditional IT management, by allowing its users to ignore infrastructure details when using services. Therefore, cloud users can access to a pool of services allowing them to load, process, create and share resources regardless the memory

or storage capacity of host devices which are managed by the cloud provider. On the other hand, SOA makes use of a set of properties targeting heterogeneous and independent systems interoperability. In this case, providers expose specific applications, in the form of services, which can be (re)used through the Internet and integrated in the information systems of clients with minimal effort of coupling. Although services based on SOA can exist independently of the cloud, there is a strong relationship between them. While SOA focuses in the architecture and design to facilitate the integration of business processes, the cloud offers the IT resources for storing and computing the functionality of those business processes. Several authors [71] [72] have stated the co-dependency between these two technologies by affirming that services published in the cloud need a SOA support behind them. In this work, I assume that both SOA and the cloud share the notion of service at the core of their technologies. Therefore, by referring to the term "service-based" approaches, both SOA and the cloud are encompassed.

In spite of the clear advantages of services, security has been considered by several authors [33] [74] [14] as a critical aspect limiting the widely applicability and implementation of the service-based technologies. Indeed, this new paradigm of communication has highlighted new security weaknesses which are added to the well-identified vulnerabilities of traditional information systems. Security continues to be an active area of research and not a few approaches have been proposed to tackle the emerging vulnerabilities of these technologies. In particular, talking about security in services traditionally implies referring to issues that arise around authentication and authorization [43] [100], integrity of data [9] [2], auditing [92] [50] and unlinkability [8]. Thus, security has been addressed by guaranteeing some properties over two main resources, namely, the web service, as the "materialization" of a business process, and the data acting as input/output of the web service execution.

I will show throughout this dissertation that this vision of security leaves an important aspect aside, which is the control of the organizational resources facing to intentional or unintentional misbehaviors of the external organization. This loss of control is caused by the fact that during the provision of a service (a business process) some resources are exchanged between clients and providers, and moved out beyond the domain of the controller. This work proposes a controllability method to address this problem. Such an approach requires to clearly define what the actor, the behaviors and the resources involved in the intended control are. It is in fact, a complex scenario relating service-oriented technologies, security, governance, risk management and even legal provisions. The present contribution considers those research areas with a new approach focused on the behaviors of the service actors regarding the usage of organizational resources.

1.2 Motivation

Social networks, the cloud and Internet of Things are only a few significant examples which show the need and usefulness of "being connected". Business dynamics have not passed up this trend and have used the advantages of the Internet to incorporate new paradigms of communication in the core of their business logics by publicly exposing functionalities in the form of services. Thus, for

both individuals and organizations the importance of service-based technologies lies in their ability to offer an alternative to traditional information systems. For organizations, it has represented an opportunity to find business partners, expand their markets and generate competitive advantages. For individuals, the service-based technologies have favored an easy access to resources, while developing new paradigms in which the technology is used as a tool to carry out activities. Final users have therefore changed their way of buying goods, share information, store data or even personalize the content they access. The range of services is large and almost any activity supported by an information system can be exposed as a service.

Service-based approaches have shown their importance for supporting business goals and communication needs, which are reflected in the rapid and wide growth of these technologies. According to Companies and Markets, in 2011 the SOA market grew 24% faster than expected over its projections, representing a business of $5.518 billion. Enterprises such as SOA Software showed a growth of 90% over 2013. Moreover, ResearchMoz predicts that the SOA market will reach $16.4 billion by 2020, which will represent a growth rate of 10.35%.

Regarding the cloud, the picture is far from discouraging. According to Forbes "in 2016, spending on public cloud Infrastructure as a Service hardware and software is forecast to reach $38 billions, growing to $173 billions in 2026. SaaS and PaaS portion of cloud hardware and infrastructure software spending are projected to reach $12 billion in 2016, growing to $55 billions in 2026. [...] The worldwide spending on public cloud services will grow at a 19.4% compound annual growth rate (CAGR) from nearly $70 billions in 2015 to more than $141 billion in 2019".

This data attests the impact of service-based technologies at both IT and economy levels. As Yefim Natis, VP at Gartner Inc. said "if you don't yet know about SOA and how it will change your enterprise's IT architecture, you are placing yourself at a competitive disadvantage". On the report of Accenture Technology Vision 2016 "People First: The Primacy of People in a Digital Age", SOA and the Cloud are presented as complementary approaches, both holding a new corporate cultural shift, where organizations are built for change. As a result, there is a strong commercial and academic motivation to research in this area, offer solutions to current limitations and leverage its advantages. It is not only an economical issue but also a new paradigm which changes the way of both doing business and providing functionalities.

1.3 Problem Statement

From the Business-to-Business (B2B) point of view, the inclusion of external services in the core of the business processes represents interesting managerial and organizational advantages. Requesting the provision of an external service means delegating a process that was before in the domain of the organization to an external provider, by also delegating the resources[1] needed for the execution of that process externally. Nevertheless, once those assets are shared/delegated for the provision of the service, there is **no control** on the way how the external partner uses them. In this context, an asset is defined as any resource (device, data, activity,...) with a value for the organization. Similarly, the usage refers to

[1] Hereinafter referred to as *assets*

any coarse-grained **business activity** in which the asset is directly or indirectly involved.

In collaborative environments, sharing assets is an unavoidable but risky condition since the consequences of such loss of control are linked to the damages that may arise due to the intentional or unintentional misuse of assets, i.e. due to the performance of an unwanted activity such as loss of reputation, loss of customers, legal actions, loss of competitive advantages or penalties. Indeed, even if service partners collaborate to accomplish a goal, they could have **opposed objectives**, which are reflected in competing interests between clients and providers.

As regards the loss of control on data, current methods for Data Loss Prevention (DLP) propose a holistic approach based on three components, namely: data governance, data loss prevention controls and support for information security process. Although DLP is a comprehensive approach which considers policies, legislation, quality and risk assessment issues, the implementation of the specific DLP controls deserves a deeper analysis. Indeed, they are focused on preventive controls, where the concrete implementations are specified according to the state of the data; for example, for data in use, controls such as usage monitoring, user monitoring or use of test data are proposed. The analysis of these specific controls in the light of my target problem let us to conclude that most of the existing approaches aiming at the protection against resources leakage are not applicable to service-based technologies. First, the disclosing of resources is not avoidable; what is more, sometimes more the information is disclosed, better results are got. It means, to consider a wider and constantly changing perimeter of security due to the dynamic nature of the service-based approaches. Second, if the asset is in the domain of the external partner, monitoring techniques do not represent a feasible solution for all assets due to the principle of technological independence and transparency. It means, partners involved in a service provision should be considered as black boxes, and neither can the external organization be controlled, so fine-grained usage actions are not observable. Third, because apart from data, other kind of assets need to be considered within service-based approaches.

On the other hand, more specific methods have been proposed in the context of the service-oriented technologies aiming at the protection of organizational resources. Even so, talking about protection undoubtedly refers to security, and so to the traditional security properties: confidentiality, integrity and availability, which are rethought for tackling the challenges of the distributed scenarios. The main limitation of those approaches is that their backbone is the web service, which means that some aspects of the interaction are not considered, in particular, the behavior the external partner regarding the assets when they have left the security domain of the controller.

To summarize, most of the current approaches assume that the asset remains within the domain of the organization which controls its use, where some privileges can be granted to external partners, allowing them to enter that domain. However, they do not guarantee some properties over those assets after they left the domain of control. Moreover, they offer a limited definition of control addressing fine grained actions, observable at the information system level, such as write and read. Finally, to the best of my knowledge the only asset covered by those approaches is the data itself. Consequently, and supported by the literature review presented in the following chapters, I identified that there is

lack of methods aiming at the control in the use of assets, addressing a larger set of activities and assets. I will refer to it throughout this dissertation as the problem of the lack of controllability. In order to illustrate the set of assets and activities covered in this dissertation let us consider a merchant who ensures the quality of his products with the label "Made in France". Since such a merchant outsources the fabrication of some of his inputs to an external organization, he needs to be sure that those pieces and raw materials are of French origin. However, he cannot directly control the internal processes of his providers or the suppliers of his provider. So he needs to be able to include this requirement as part of the terms governing the service provision, to represent it in a machine-readable form to be automated, and to guarantee that it is respected. Thus, he can prove that he behave correctly and probably can be exempted from liability in case of complaints. Like this, there are many examples in the Business-to-Business environments, in which the guarantees about the service provision are defined in terms of more high-level organizational requirements than only data security or performance, and where the business activities associated to those requirements are restricted to the fact that the asset is out of the domain of control. For such organizational requirements, the scope and expressiveness of the current machine-readable service agreements are limited.

1.4 Objectives and Contribution

As it was mentioned above, in the service-oriented approaches, organizations face situations in which they can neither supervise nor guarantee the use of the assets when they are out of their domain of control. To the best of my knowledge, little research exists on that matter when a large set of assets and more-coarse-grained activities are considered. This dissertation has for objective to propose a method which supports controllability in the following way:

- To choose a machine-readable representation able to formalize business guarantees about the provision of a service.
- To present a formal definition of controllability as part of the non functional business requirement governing the relation between service clients and providers.
- To propose a model for the controllability requirements which be able to represent the expected behavior of an external partner regarding the use of assets.
- To verify the compliance with the controllability rules in order to evaluate the performance of the service actors and the quality of the provided service.

Considering the aforementioned objectives, this work presents as novelty and main contributions:

- A generic model which formalizes the semantics of the elements involved in the contractual relation entered into the client and the provider of a service.

- A formal definition and concrete method to address controllability requirements. This contribution improves a previous work [68] in which controllability is defined in an abstract form as a security property.
- To propose machine-readable service contracts to extend the scope and expressiveness of SLAs. Such a contract improves the SLA guarantees, which are traditionally based on security and performance by defining restrictions about the behavior of the service actors regarding the use of assets.
- A method for the assessment of the service actors based on the available knowledge about their proved behavior.

1.5 Thesis Organization

This document is structured in three parts as follows:

- **Chapter 1 : Introduction**
  In this first chapter, the perimeter of the dissertation was delimited. The aim of this chapter was to establish the frame of my proposition by narrowing down the specific problem to be addressed as well as the objectives pursued with this research.
- **Part I - Framework**
  - **Chapter 2 : Background**
    In the second chapter, a synthesis of the approaches related to the problem tackled in this dissertation is presented. The aim of the described background is twofold. First, it builds the conceptual foundations that allow to have a clear understanding of the terminology and methods used in the subsequent chapters. Secondly, it presents the panorama that justifies the methodological and design choices made to address the proposed objectives. However, as it was previously mentioned, the problem of loss of control in service-oriented approaches impacts several domains such as security, representation of non-functional service guarantees and accountability and legislation; therefore for the sake of simplicity, the state of the art is split between Chapters 3, 4 and 5, which allows to have a better comprehension of the specific contribution presented in each chapter with regards the limitations of the current works.
- **Part II - Models for Controllability based on Contracts**
  - **Chapter 3 : Semantic-based Service Contract Model**
    The findings of the first objective of this dissertation are presented in Chapter 3. I formalized the components involved in the controllability of organizational assets through a semantic model of contracts, supported by the $\mathcal{SROIF}(\mathcal{D})$ representation. At the end of the chapter, a validation of these knowledge formalization is done by the creation of a machine-readable contract in Web Ontology Language (OWL).

  - **Chapter 4 : A Model for the Controllability Terms based on Rules**
    After obtaining a clear definition of the elements involved in my specific domain of knowledge, let us offer a model which makes use of those defined elements to represent requirements about the expected use of assets in the form of rules. Similarly, the formalization of evidences is added to the model in order to guarantee the compliance of the external organization with the established policy. Analogously to the previous chapter, a validation of the proposed usage conditions is presented with the creation of policies in XML-format.

- **Part III - The Process of Asset Controllability**

  - **Chapter 5 : Knowledge-driven Reasoning on Controllability**
    Throughout Chapters 3 and 4, the concrete implementation of the controllability model is done by its machine-readable representation in XML-format. In Chapter 5, I demonstrate the usefulness of this concrete representation by carrying out a (semi-)automatic process which takes as input the proposed contract and a log recorded during the service provision to make inferences about the compliance with the contract and to evaluate the performance of the actors involved within the service provision.

- **Part IV - General Conclusion**

  - **Chapter 6 : Conclusion**
    In this Chapter, the main arguments developed during this work are summarized and the contributions presented throughout the other Chapters are discussed in the light of the aimed objectives. In Section 6.2 the issues opening the door to further research close this work.

Part I

Framework

Chapter 2

Background

The service-oriented technologies are based on a set of principles and characteristics enabling the successful interoperability of systems. In this chapter, before detailing the controllability method, I describe the main building blocks of the service technologies. This chapter does not intent to cover in depth all the service-related issues but to introduce the main foundations supporting and guiding my proposition.

2.1 Service Oriented Architectures (SOA)

Currently, SOA is a frenzy and promising technology implemented to leverage collaborative business goals facing the complex and rapid enterprise changes imposed by market tendencies. From the Business to Business (B2B) level, SOA is an architectural paradigm which allows to use and integrate external applications, called services, within internal business processes of an organization. From the technological point of view, this collaboration is tackled through the interoperability of the information systems that hold the business logic of each organization. Such an interoperability is based on a minimal coupling which gives it a dynamic nature and rapid adaptation to changes.

Although, SOA and web services[1] are both buzzwords, they are not new concepts at all. Several distributed architectural models such as DCE, RPC, ORB, CORBA, DCOM and RMI had been already proposed to facilitate the communication between systems. However, they did not have the enormous popularity of SOA, because of their lack of standardization in technologies and their inability to support heterogeneous environments and composite applications [33] [91]. SOA, on the contrary, is based on standard object representations which facilitates the integration of system regardless the platform or implementation language used by the partners involved in the service provision.

In general terms, several properties are associated to an architecture based on services:

[1]Even if SOA is not tied to web services, it is currently the most used application.

- *Reuse*: services are implemented having in mind their reutilization. Thus, they are deployed only once and could be continuously accessed within the organization or by external clients.
- *Loosely-coupled*: services, as complete functional applications, are independent from each other and from legacy applications.
- *Transparency*: internal service functionality and changes in the implementation remain transparent for the user of the service.
- *Integration*: services have to be easily integrated to existing business logics and have the ability to cooperate with other services in order to build complex business processes.

To attain the integration between heterogeneous systems while keeping their loosely-coupled and transparency characteristics, three architectural components are required, which, in the case of SOA, take advantage of standards based on eXtensible Markup Language (XML) to fulfill the requirements of communications and interoperability. Such components are:

1. Communication protocols

   To leverage the message-based communication capabilities of SOA, the format and protocols of the exchanged information are also standardized. The service oriented-architecture uses of HTTP, FTP and SMTP to send messages over the Internet. Those messages could be written following a SOAP or REpresentational State Transfer (REST) format. The popularity of both formats differs from one reference to another, and both prove advantages and shortcomings. Notably, their comparison criteria are defined in terms of support to security specifications, technologies (browser and languages used for data representation), tolerance to errors, simplicity and efficiency.

   Particularly, a SOAP message is referred to as an envelope, which is composed of two main components: an optional header and a mandatory message body. In the header, some kind of meta-information about the message is included, which becomes one of the main advantages of SOAP compared to REST. This header is of great importance because it allows such information not directly belonging to the service functionality travel in the message, for example security information. On the other hand, the body of the message contains the actual data exchanged between the service client and the provider. Figure 2.1 shows an example of a SOAP message.

2. Service description

   Web services make use of standard Internet protocols such as FTP and HTTP to be published, discovered and accessed by clients. In order to carry out these activities, web services need to expose a minimum set of information, enough for allowing its use, without exposing the details of its internal functionality. Web Service Definition Language (WSDL) is a W3C recommendation to create public interfaces which indicate how to use the web service. This language allows the definition of a minimum set of functionality information (data types, methods, inputs and outputs)

SOAP Request

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:toSayHello xmlns:ns1="http://samples.wrap">
      <firstName>L</firstName>
      <secondName>Pipe</secondName>
    </ns1:toSayHello>
  </soapenv:Body>
</soapenv:Envelope>
```

SOAP Response

```
<?xml version='1.0' encoding='utf-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:toSayHelloResponse xmlns:ns1="http://samples.wrap">
      <return> L say hello to Pipe</return>
    </ns1:toSayHelloResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 2.1: Example of a SOAP Message

as well as protocols and messaging formats. WSDL files specify the messages exchanged when using the service, endpoints to access published operations and the protocols used to enable communication.

3. Service discovery

   This component of the SOA architecture acts as a service registry, by allowing to publish, search and discover available services. The most common registry is the Universal Description, Discovery and Integration (UDDI), which is a recommendation of the OASIS group. The UDDI receives XML requests from the service client concerning a requested service and returns a *.wsdl file containing the description of the service that fulfill the client specifications.

2.1.1 Composed Services

When the complexity of a required service cannot be satisfied by a single provider, more complex paradigms are involved. The composed services use their integration property to collaborate with other independent services in order to satisfy a required business process, which is commonly modeled as a series of steps in a workflow. An interorganizational workflow is defined as a cooperation of several, autonomous and heterogeneous business processes, in order to achieve a common goal [73]. From a design perspective, workflows represent the coordination of several tasks to accomplish a business process. An example is flight booking companies, which in addition to the airline companies, collaborate with other organizations to provide optional services such as hotel and car reservation.

Composed services are divided into two approaches: choreography and orchestration. The term choreography has been coined through the analogy with dancers “who dance following a global scenario without a single point of control”. Thus, in this approach, no central entity coordinates the workflow and so

each service knows the whole business logic and the messaging sequence (Figure 2.2). Choreography-based services are abstract models, and therefore they are not executable. However, some approaches (the most well-known are WS-CDL and WSCI) have been proposed to define the rules that enable the service collaboration. Currently, none of these approaches is supported by any standard.

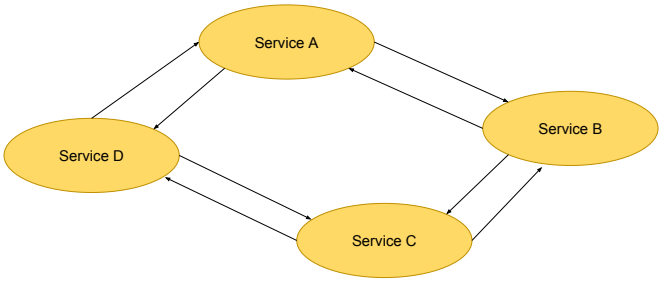


Figure 2.2: Services Composed with a Choreography Approach

In the service composition based on an orchestration approach, a new service composed of existing ones is exposed and executed in a business process engine. In this approach, a central entity named orchestrator coordinates the whole service provision, i.e., it knows the services intervening in the business process and the messages exchanged. Business Process Execution Language (BPEL) is the standard language used to describe orchestrated services. This language creates WSDL descriptions of the composed services, which could be executed in engines such as Apache ODE.

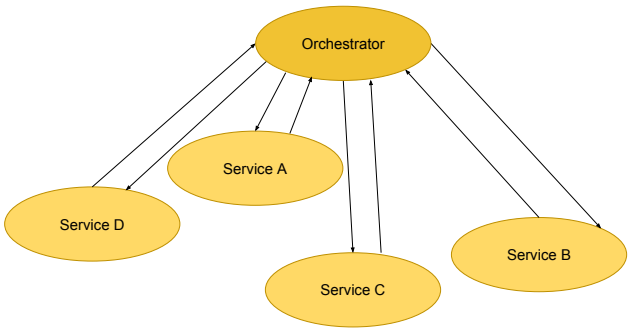


Figure 2.3: Services Composed with a Orchestration Approach

2.1.2 Semantic Web Services

The Semantic Web's main motivation is the evolution of the Web, by moving from the access to a large collection of information dedicated to the human analysis, to a smart web delivering filtered and relevant information to users. To do that, the semantic web relies on machine-readable annotations capable of describing the semantic of the data published on the web.

Recognizing the importance and widespread use of the web services in both Business-to-Client (B2C) and B2B approaches, the semantic web services take advantage of the principles developed in the semantic web to automate the discovery, negotiation, composition and invocation of web services. Indeed, WSDL descriptions are limited in their representation since they focus on describing functional aspects about how to access and secure the service (port, endpoint, expected inputs, Service Level Agreement (SLA) and security policies); however other specifications associated to what the service does and how they interact and collaborate with other services are written in natural language, which implies an explicit human intervention and processing. By integrating semantic annotations understandable and processable for the machine, the gap between current definitions based on syntactic representations (XML) and the semantic behind these representations will be bridged, becoming the service provision a smarter process.

Figure 2.4 [38] illustrates the Semantic Web Services as the intersection between the evolution of the Web and the evolution of web services to a dynamic view.

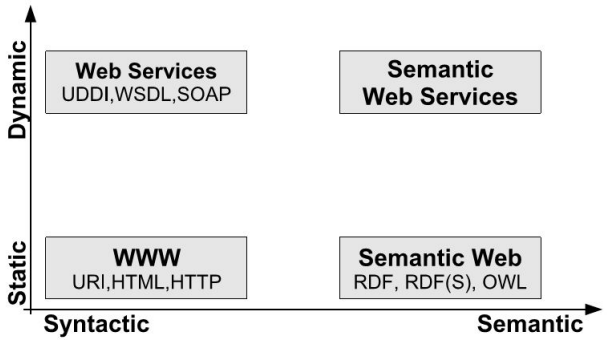


Figure 2.4: Web Evolution

2.2 The Cloud

The genesis of cloud computing seems to be a natural evolution of IT technologies regarding organizational needs. Not long ago, companies had their own IT division in charge of managing, developing and giving solutions to existing organizational requirements, just as companies implemented new business models, the complexity of IT solutions increased as well as the financial and human capital investments required to give support to the growing and challenging changes. In this scenario, organizations moved a part of their business processes, goods and assets to a third entity known as cloud service provider. Thus, Cloud computing uses the Internet to allow an on-demand access to a pool of resources. The basic idea behind Cloud computing is to separate physical components (managed by a Cloud provider) from virtualized ones (managed by Cloud users). This allows the cloud provider to move data and applications freely toward several physically distributed hardware. The main attractive features of the cloud are associated to reduce IT costs, scale up/down due to its

elastic nature, ubiquitous access, offering on-demand services and support to pay-per-use approach.

Since its beginnings, the cloud has been associated to three types of service models, namely: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). However, as the Cloud grew in popularity, emergent needs showed up and thus two new layers are commonly referred to as being part of the Cloud: the Business Process as a Service (BPaaS) and the Content as a Service (CaaS). All of these layers represent different types of services offered by Cloud providers through the Internet. From bottom to top in Figure 2.5, the cloud offers to their users access to servers and network resources belonging to a cloud provider (IaaS), support to platforms allowing to deploy applications on the provider-side (PaaS), access to software hosted by the provider (SaaS), customized access to content based on client's requests and interests (CaaS) and outsourcing of services provided by third parties (BPaaS).

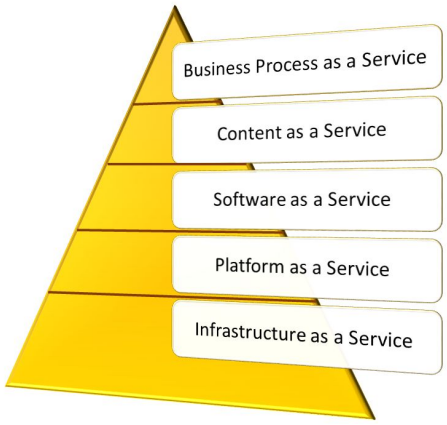


Figure 2.5: The Cloud Layers

2.3 Service Level Agreement (SLA)

A SLA represents an agreed document in which requirements about the quality of a service are established. In the context of web services, SLA often refers to the performance and security properties of the service. In general terms, the life-cycle of a machine-readable SLA is composed of a six-step approach, namely: template definition, publishing and discovering, negotiation, implementation, execution and monitoring. Each of these steps represents specific challenges either from the business or technological point of view. For example, the definition of the Quality of Service (QoS) parameters is not an easy task due to the difficulty to formalize the criteria under which the service will be evaluated. Similarly, for composed services, the SLA template definition should consider the separation of duties for each partner, a more complex monitoring of the QoS parameters, the construction of the business workflow, and negotiation strategies, specially in case of decentralized service provision [109].

Several machine-readable SLA (mainly based on XML) have been proposed in the literature, such as SLANG [69], WSLA [75], WSOL [108] and WS-Agreement [5]. Most of the existing approaches are modifications of two of the most accepted specifications, i.e., WS-Agreement and WSLA [75], including new aspects such as the number of signatory parties or new monitoring techniques. However, as shown in Figure 2.6 for the WSLA model, current service agreement approaches usually link the definition of the SLA parameters to a metric. So, due to their measurability characteristic, most of the current SLAs only define performance guarantees at the infrastructure level (hardware availability, power availability, data center network availability, backbone network availability, service credit for unavailability, outage notification guarantee, Internet latency guarantee, packet loss guarantee) as well as a minimum set of business levels including penalties and payments. Despite its limited expressiveness, SLA is an important reference for this work, allowing to identify the modeling and machine-readable representation of non-functional service requirements. The state of the art as well as alternative representation approaches are presented in Section 3.2.

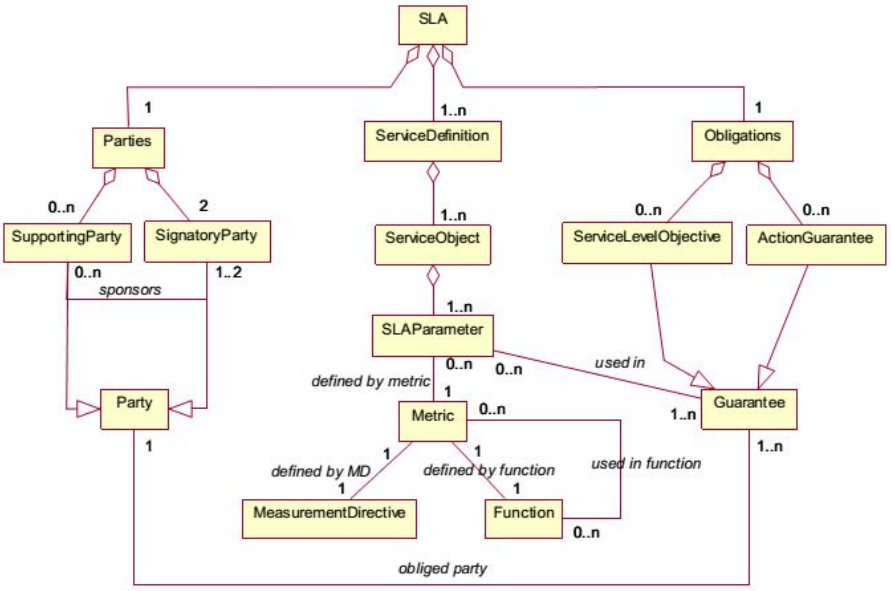


Figure 2.6: WSLA Model

2.4 Governance

In general terms, governance refers to the processes regulating the elements involved in a specific system for the successful accomplishment of a goal. The governance principles can be applied to a city, a country, an organization or a project. As depicted in Figure 2.7 [16], governance is defined from the interaction of three components: people, policies and process, in order to obtain a

desired behavior. People are considered as the main factor contributing to the success of governance since they are responsible for establishing the set of rules, the policy, leading to the desired goal. The governance processes comprise the strategies which ensure that the established policy is respected. All together, people, policies and process result in the accomplishment of the desired behavior.

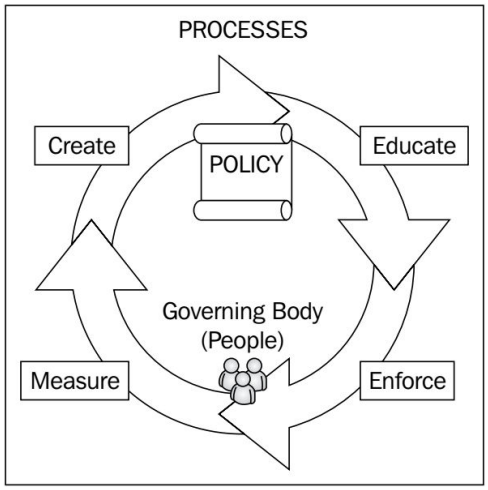


Figure 2.7: Governance

In particular, IT governance aims to fill the gap between business needs and IT management by making decisions first about what projects deserve to be funded, and second about the correct management of funded projects. In the literature, three main frameworks are identified addressing IT governance, namely, COBIT [54], ITIL [11] and the ISO 27002 [55]. In the model of IT governance of COBIT 5, the information and the technological assets are seen as the key for the generation of organizational value, whose governance is enabled by the application of five principles:

1. Meeting stakeholder needs
2. Covering the enterprise end-to-end
3. Applying a single integrated framework
4. Enabling a holistic approach
5. Separating governance from management

More concretely, the SOA governance field ensures that the organizational efforts in terms of the service oriented architectures are successfully implemented within the organization. Indeed, as stated in [16], SOA is not a technological matter, it is about "changing the behavior of the organization so that it can improve". Thus, the governance is based on the principle that the service-oriented approaches do not mean to translate legacy applications into new ones XML-based functionalities, but that their implementation should respond to

some organizational need, for example, in terms of money saving, collaboration or interaction. Consequently, people, policies and process should be addressed synergistically around that desired behavior which is the organizational motivation for the SOA implementation.

In this dissertation, the great relevance of considering a governance perspective for controllability is highlighted. Clearly, an approach aiming at the control over assets requires an analysis from a three-dimensional perspective. First, people leading (responsible for) the definition of a desired behavior with respect to the usage of assets; second, concrete policies reflecting those needs; and finally, specific processes allowing to evaluate if the controllability policy is respected or not. In [58], the first stages of what would become the current proposition are presented, where the control over data is analyzed as a multidisciplinary issue involving legal, social, technical and organizational aspects. Moving on from such analysis in the light of governance, social and organizational aspects are easily covered into the "people" component, while "policy" and "process" components are covered by technical and legal aspects.

2.5 Normativity about the Use of Assets

As proposed by the first principle of COBIT 5, so as to meet stakeholder's needs, an organization needs to be aligned with external regulations[2] as an important element of risk management. In the interaction between service clients and providers, public legislation is presented as a regulatory framework dealing with policies that restrict and regulate data processing, by aiming at data protection, data loss, data retention and accountability.

Currently, information systems gather increasing amount of data from different sources. The growth of social networks and the easy way of exposing personal information suppose stringent concerns about privacy, ethics and even computer crime. Indeed, by using the appropriate techniques it is possible to analyze and merge different sources in order to link personal information. This concern motivates the creation of policies targeting the way how the data should be processed.

The European Union has played an active role regarding data protection by offering several directives and regulations clarifying the responsibilities of each data stakeholder [34], [36], [53], [35], [37]. Particularly, the following definitions are coined:

- *Controller*: "the natural or legal person, public authority, agency or any other body which alone or jointly with others determines the purposes and means of the processing of personal data". Controllers can, therefore, collect and process personal data. They are responsible for describing in detail the nature of the data, the purpose of their processing, monitor data processing and establish the required quality of service.

- *Processor*: "it means a natural or legal person, public authority, agency or other body which processes personal data on behalf of the controller". Data processor is responsible for naming all the sub-contractors that it uses for processing the data, and implement all the technical and organizational

[2]Regulations defined externally to the organization such as national or international laws.

measures that allow the compliance with the established contract (private accountability) and data protection legislation (public accountability).

- *Third Party*: "it means any natural or legal person, public authority, agency or any other body other than the data subject, the controller, the processor and the persons who, under the direct authority of the controller or the processor, are authorized to process the data".

- *Recipient*: "it means a natural or legal person, public authority, agency or another body, to which the personal data are disclosed, whether a third party or not".

On the other hand, a different approach is proposed in the United States regarding data exchange and privacy. Although not a few US acts exists, the most known one is the Uniting and Strengthening America by Provided Appropriate Tools Required to Intercept and Obstruct Terrorism Act (USA PATRIOT Act) [104], which grants the US government the right of accessing to personal and commercial information. A comparison of the US, EU and Australian regulation is presented in [105] for the particular case of transborder data in the Cloud. That work is based on the differences between identification and [transaction] identity. The authors state that given the characteristics of the cloud, more efforts should be done regarding regulations about data disclosure, such as those in Australia, and not only about data transfer.

In terms of data legislation and protection, other laws exist around the world such as those in Argentina [115], Chile [30] and Japan [85], all of them targeting the regulation of data collection and processing. Irrespective of the legislation of each particular country, an analysis of the data, and especially that data belonging to the category of confidential and personal information, is presented in [94] in the light of the property right in data. This work states that an approach based on trust, where the parties have a common understanding of their obligations, prevent long privacy policies. In this context, the controller is aware of the risks of loosing data.

Legislation and its enforcement have been an important issue in service provision. The main concern, as confirmed in the ICO's reports about policy enforcement [86], is that even if directives exist, they remain on paper, i.e, they are not integrated into the processes of the organization, which may result in organizational damages due to both the monetary penalties and the consequences of the disclosure of data. Although, the aforementioned directives are focused on personal data, they are in fact very useful for my controllability approach. First, they allow to identify current non-functional requirements about the use of data, considered the latter as a particular case of organizational assets. Second, even if laws might differ widely from one country to another, the above references establish clear definitions about the stakeholders involved in the processing of data. Such definitions can be used as a baseline for the identification of the elements associated with asset controllability.

2.6 Security in Service-oriented Environments

As set by the legislation, data stakeholders should also implement technical security measures in order to guarantee the processing of data. In [34] it is stated

that those measures should be consistent with the risk of the processing as well as the nature of the data. Nevertheless, the meaning of security has raised to not a few discussions due to the subjective view of what a secured process represents. According to Microsoft, security "is fundamentally about protecting assets. Assets may be tangible items, such as operations or your customer database - or they may be less tangible, such as your company's reputation."[25]. The ISO vocabulary [57] defines security in terms of the preservation of confidentiality, integrity and availability of information, where some other properties such as authenticity, accountability, non-repudiation and reliability were thereafter added. However, aiming at those properties is a challenging issue in distributed environments such as the service-oriented ones, where additional requirements need to be considered, specially, the use of standards which leverage interoperability. Taking in mind the particularities of the service architectures regarding security, several specifications and standardization efforts have been proposed mainly by OASIS, W3C, WS-I, IETF, IBM, Microsoft and VeriSign, forming what is known as the WS-* family (see Figure 2.9 [52]):

- *WS-Security - SOAP Message Security*[3]: currently in its version 1.1, WS-Security focuses on the integrity and confidentiality of the SOAP message, as well as the possibility of sending security tokens. It offers a wide flexibility regarding the format of the security token, domain of trust, signature and encryption technologies.

- *WS-SecurityPolicy*: In the version 1.3 published in 2012, a policy is defined as the way of expressing constrains and requirements about security assertions, in particular those features provided by the SOAP Message Security, WS-Trust and WS-SecureConversation. That is to say, it provides the information required to enable a secure message exchange such as information about encryption, signature, binding assertions, binding properties and token properties. WS-SecurityPolicy aims compatibility and interoperability.

- *WS-Security - Username Token Profile*: This specification aims at the identification and optionally the authentication of a requestor through the use of tokens.

- *WS-Security - Kerberos Binding*: It specifies how to encode Kerberos tokens in the SOAP messages for authentication purposes.

- *WS-Federation*: Taking in mind that the aim of federation is to widen the security domain, it becomes an interesting approach for considering control on resources as a security property. Indeed, WS-Federation aims at sharing some resources belonging to a realm A, to another realm B based on identity, authentication and authorization assertions. However, the main drawback of this specification is that it considers only access (as interaction) to data (as resource) mediated by messages. Figure 2.8[4] shows three examples of federated scenarios.

[3] http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-errata-os-SOAPMessageSecurity.pdf

[4] Figure taken from http://docs.oasis-open.org/wsfed/federation/v1.2/os/ws-federation-1.2-spec-os.pdf

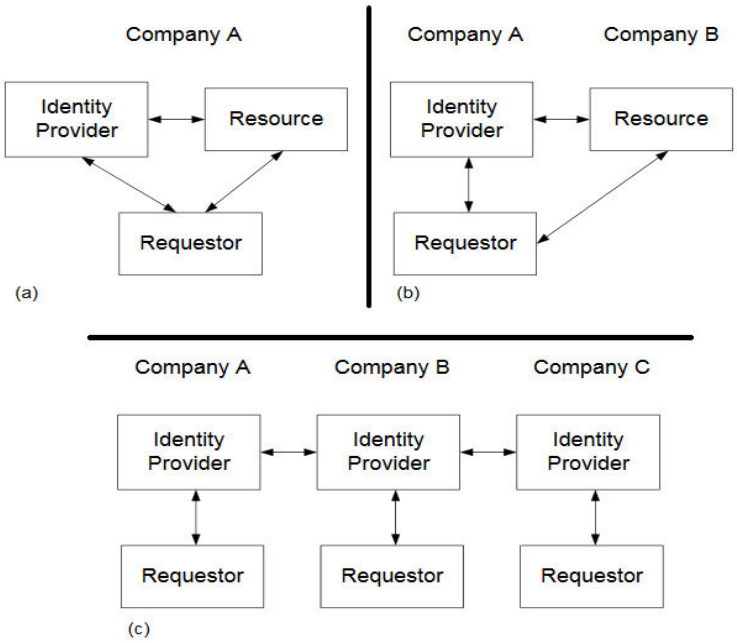


Figure 2.8: Examples of Federated Scenarios

- *WS-Trust*: This specification is one important building block for the federation of different realms. WS-Trust allows the sharing of credentials through different domains. More concretely, it focuses on determining if a party belonging to a domain can trust the security assertions of another domain. WS-Trust provides methods to issue, renew and validate security tokens.

- *WS-Security - SAML Token Profile*: This specification uses SAML (Security Assertion Markup Language) assertions as part of the SOAP message security to bind subjects and claim security assertions.

- *WS-Security - x509 Certificate Token Profile*: It specifies the way in which the standard X.509 is used in the SOAP messages through its integration within the SOAP Message Security specifications.

- *WS-SecureConversation*: It aims to secure the message exchanges by creating contexts which cover several SOAP messages. The specification defines a context as an "established authentication state and negotiated key(s) that may have additional security-related properties".

InnoQ [52] has compiled all the WS-* standards and specifications, by 2008 it was summarized around 60 solutions which have been grouped in the following specifications: messaging, metadata, security, reliability, resource, management, business process, transaction and presentation.

In general, security-related specifications target properties such as authentication, authorization, confidentiality and integrity of the data exchanged via SOAP messages, as well as the message itself. However, as in the case of this

proposition, I could argue that once a trust and secure communication with another partner is created, I may be also interested in knowing the way in which the partner uses the shared resources in its internal processes after they have been voluntarily shared. It points to a layer on the top of the SOAP messages. The proposition presented in this work goes in this direction; my model and method of control over assets does not pretend to replace existing specifications but to complement them with a higher level of abstraction. Consequently, it will be absolutely feasible (and preferable) that other specifications such as WS-Trust be thereafter added to my proposition.

2.7 Controllability

Motivated by the fact that current norms of risk management (in the context of information security, in particular the ISO/IEC 27005:2011) do not consider the notion of service, in [68] the term controllability (it was the proposed translation by authors in [67] from the original French term maîtrisabilité) is proposed as a new abstract security property at the same level than confidentiality, integrity and availability. The author defines controllability as the ability of "ensuring total control over the services used" by allowing the information system architect to qualify the level of trust on services. In that proposition, however, the actual controllability implementation is restricted to data, and more concretely to the content of the documents used on the service provision. Aiming that control, authors propose the use of metadata attached to the documents (creating hence self-protected documents following the DRM model) in order to make the traceability of the communication. In its simple form, the principle behind that proposition is to create policies embedded in the documents exchanged among the service stakeholders. Those rules govern the access to documents according to contextual information, while some kind of metadata is collected in order to calculate indicators useful to supervise the information system (Figure 2.10). In general, the initial assumption of the [document's] controllability is that the information security in service-oriented approaches is not guaranteed due to the loss of confidentiality and integrity, so an architecture based on intelligent documents is required.

The present dissertation is aligned with the above definition of controllability, as well as its underlying motivation in terms of organizational risks. However, unlike existing works [67] [79] [80] [81] I am not restricted to data, but I consider more general assets. In my approach, the result of the service provision is not so important as the way to obtain this result in terms of the use of assets (behaviors of the service stakeholders) so as to avoid organizational damages. My controllability approach aims at the enforcement of the service provision following some business rules consistent with the risk management process of each organization. Moreover, the current approach of controllability is improved through formalization, and its implementation in a machine-readable form, of the policies governing the relation between the service stakeholders. This aspect has been left as future work so far. Finally, as opposed to the current approach, the present work is not based on web service security standards, which are considered as a limited expressiveness to representing business policies (Chapter 3).

2.8 Discussion

Despite the optimistic growth of service-oriented approaches, recent studies have shown a strong fear of companies and a feeling of insecurity to deliver their assets to an external organization. In this way, their attracting advantages of SOA and the cloud have been constrained by several issues which have detracted the widely adoption of this model, namely:

1. From the legal perspective, even if norms and acts exist for governing the processing and use of data by external stakeholders, *they are limited to paper-based documents which make difficult the automatic verification* of their accomplishment. Moreover, special attention should be paid to the fact that the legislation can drastically change from one country to another, which is an important aspect for transborder data, in particular in the cloud thanks to its flexibility property, the data can be stored in a country, accessed from another one and even processed from another country. Compliance with legislation is mandatory in the risk management process as part of the governance efforts within organizations. However, it is also highlighted that current legislation mainly aims at a particular subset of organizational assets: the data, and more concretely, personal information.

2. Regarding the responsibilities of controllers and processors, *there is a common perception that an imbalance prevails in their roles*, being the processor the active entity who establishes the terms which govern their relation. Indeed, in some countries processors are not obligated to describe their intended use on external assets, and even if they do it, the counterpart does not have the possibility to object it. A typical example which illustrates this issue is obvious in web services provided under the B2C paradigm. In such a case, long documents written in natural language, usually by making use of legal terminology, is presented to the client before accessing the service ("click and accept" contracts), however, those terms are not negotiable and the client is compelled to accept them in order to access the service.

3. During an external service provision, organizational assets move freely between different business domains, where the controller has not guarantee about their actual use. Although approaches such as the creation of SLA documents exist to establish some criteria governing the relation between clients and providers, *those current models have a limited expressiveness to represent business restrictions on the service provision*, in particular, regarding the use of resources belonging to the controller.

4. Finally, the assignation of responsibilities is also an important matter for clients and providers. Some guarantees are needed to reassure clients and providers that, in case of damages, they will be compensated. Moreover, as established by the legislation, evidences are also needed to impute guilt to a stakeholder, or on the contrary, to release him from liability. However, such guarantees and evidences are subjected to a dichotomy requirement. On the one hand, *enough information is required which allows to link actions and activities to liable stakeholders*. On the other hand, that information should be *collected considering the legal framework and without*

exposing the internal business logic of the organization. Moreover, this challenge is *accentuated in composed services* since it is difficult to detect faults and to establish a chain of responsibilities due to the fact that business processes may be orchestrated through several domains.

In the light of the above analysis, it is not difficult to see that a proposal targeting the "behavior" of the service stakeholders in terms of business expectations about the use of assets involves the taking into account of several approaches such as legislation, social impact of assets misuses, risk management, assessment of a service based on the expected service quality and liability in case of damages.

As regards to the information allowing to have guarantees during the interaction with external partners, several approaches are identified in the literature, which can be grouped into three categories: user-centric (which aims to monitor the actions carried out by each partner belonging to a workflow), process-centric (which aims at the monitoring of the workflow itself), and data-centric (which monitors the data along the workflow). Each of these approaches has its advantages and drawbacks; however recording some information which allows to make decisions about the service provision seems to be a common denominator among them. Regardless the approach implemented, several issues have to be considered. First, due to the fact that penalties could be imposed to faulty entities, some of them had rather hide their faults. Thus, verifying that reliable information is recorded about the service provision becomes a non-trivial concern. Secondly, keeping the business logic of an organizational as a black box for external partners is a relevant issue in order to avoid that some vulnerabilities could be exploited. In terms of monitoring processes, it means that not every single action can be recorded. Thirdly, improved algorithms are required in order to detect the origin and the cause of violations of the service terms from the data collected during the service provision. Consequently, it is clear that the collected information matches what is called as metadata in the controllability approach.

Finally, this discussion about the service-related issues is closed by stating that an approach aiming at controllability should be addressed within a governance framework. Let us consider Figure 2.11 in order to clarify the way in which the approaches described in this section influence my proposition and how the different building blocks of my approach are articulated within a governance approach.

- **Intended behavior**: The implementation of governance focuses on gaining control on assets in order to avoid organizational damages due to the misuse of assets by an external organization during the provision of a service. In Chapter 3, the elements involved in such a desired behavior are identified and formalized through a semantic model. This first contribution sets as a basis, but also improves, current approaches about semantic web services.

- **People**: From the desired behavior, the problem to be tackled is the loss of control on resources, where the controller does not know how its assets are used by external organizations. In this context, I identify three main actors: the provider, the clients and third parties. As previously mentioned, during the service provision, some assets are exchanged from

both sides of the interaction, meaning that the notion of controller and processor is associated with a particular asset and not to the service itself.

- **Policy**: In order to guarantee that the external partners use the assets without carrying out activities that could generate damages to the controller, it is needed to explicitly establish what the expected behavior of the processor is. The policy formalization as well as its concrete representation in machine-readable form will be presented in Chapter 4. The proposed policy considers state-of-the-art formalizations of non-functional requirements, in particular SLA, and overcomes its limited expressiveness by adding components which allow the definition of high-level business requirements.

- **Process**: In order to guarantee that people comply with the policy, I propose to use evidences and a log (acting as controllability metadata) which allow to assess the service provision versus the established policy. Through Chapters 5 and 6, I will present the proposed method aiming at the semi-automatic reasoning about the validity of the policy and the quality of the provided service. For the proposition of this method, current approaches addressing responsibility and QoS assessment in the context of service-oriented approaches are considered, as well as legal issues about the collection of metadata.

2.9 Conclusion

In this chapter, I have outlined the main fields approached to tackle my identified problem. Due to the fact that my proposed solution relies on a method where both model and process are described, I split the state of the art in the following chapters, which privileges the chaining of ideas between the current limitations in a specific field versus my contribution. Even so, I seek to present the main guiding principles of my approach, which justifies the design and methodological choices made in this proposition.

In general, although several open issues about the service provision, specially in inter-organizational environments, continue to receive attention by the commercial and academic community, a solution that overcomes every of the aforementioned issues is an ambitious and complex proposition. In this dissertation, I propose a solution targeting one particular limitation identified in the service-oriented approaches, namely, the loss of control in the use of assets by an external organization. Although in compliance with the general definition of controllability reported in the literature, this approach extends it by the proposition of a formal model which covers not only digital content but also any organizational asset involved in the service provision. Similarly, this approach considers a governance framework in which policies in a machine-readable form are proposed to address the lack of controllability while some processes evaluate and give feedback to those policies.



Figure 2.9: WS-* Family

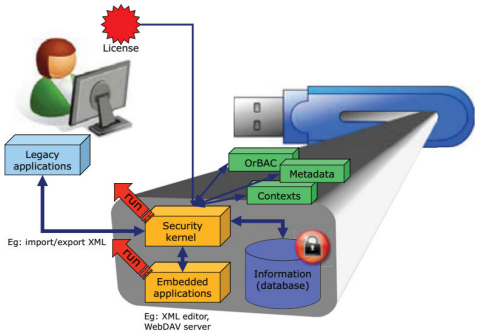


Figure 2.10: Controllability on Documents



Figure 2.11: Controllability Governance

Part II

Models for Controllability based on Contracts

Chapter 3

Semantic-based Service Contract Model

Current approaches governing the relation between service clients and providers are based on claims specifying guarantees about the service provision. However as it was argued in the previous Chapter, those claims do not cover business requirements on controllability. Taking in mind the challenges associated with the definition of controllability rules, in particular, the use of a specialized vocabulary, I propose a semantic model which formalizes the components of a service contract in this Chapter. I highlight the advantages of the Description Logic (DL) and ontologies to formalize such representation. The proposed model is validated through its representation in a machine-readable form in XML where an example coming from a real case is presented. This chapter concludes with the perspectives of service contracts, notably in terms of negotiation.

Objectives

- To choose a machine-readable representation able to formalize business guarantees about the provision of a service.
- To propose a formal definition of controllability as part of the non functional business requirement governing the relation between service clients and providers.

3.1 Introduction

In the B2B approach, as previously stated, the delegation of business processes to an external partner is nowadays a common practice mainly motivated by the reduction of costs and the need to offer added-value services. In such a collaborative relation between clients and providers, even if each organization keeps its autonomy, the fact of interacting through business activities generates organizational dependencies that could affect both parties. In order to illustrate this interdependence, let us consider the case of an online retailer who interacts with

manufacturers and sellers to offer products with several shipping companies for the product delivery. Another common example concerns flight search enterprises which need to interact with several airlines companies in order to offer the cheapest price; moreover, they collaborate with other partners to offer additional services such as car rental and hotel reservations. Besides, it is well-known that *social sharing* has been mainly leveraged by service-oriented technologies, therefore it is common to find services such as carsharing which work together with insurance companies. In general, there are quite a few examples showing the advantages from the point of view of interaction and communication of service-oriented technologies within B2B environment. Nonetheless, it should be also highlighted that during the external service provision, the provider becomes a link in the value chain of clients, which means that some behaviors on the provider side will be reflected onto clients and vice versa. That clearly represents organizational risks that both partners intend to control and reduce. Consequently, in that inter-organizational relationship it is not only important to know "who does what" (functional requirements) but also to know *"how he does it"* (non-functional requirements), in order to avoid that such dependency generates a domino effect causing negative consequences for the organization. Consider for example that a late delivery might affect the reputation of the retailer (or causing monetary losses), however the delay may be caused by the manufacturer to deliver products. Therefore, there is a need to keep control on the organizational resources by having guarantees which allow to ensure that they are used by external organizations according to some business rules.

All of this has major implications from the organizational and technological point of view: first, by defining which are the organizational resources which need to be protected. Secondly, by formally representing what they need to be protected from. Thirdly, by broadening the criteria used for the assessment of the quality of the service. For the moment, I will focus in this Chapter on finding a representation aiming at formalizing the elements involved in the provision of a service, which will serve as a basis to identify the components of controllability. This chapter named "Semantic-based Service Contract Model" is structured as follows. In Section 3.2 I will present the current approaches with respect to the representation of non-functional requirements in service-based environments. In Section 3.3, I justify the choice of using contracts to represent non-functional requirements on the use of assets. By considering contracts as the basis of the proposed representation, the suggested generic model formalizing the semantic of the elements involved in the relation between the client and the provider of a service is presented in Section 3.4. The validation of the semantic model is described in Section 3.5 by the implementation of service contracts in a machine-readable form which is illustrated with an example from the Oil&Gas industry. This Chapter will end with the discussion of the proposed representation and the conclusion in respectively Sections 3.6 and 3.7.

3.2 State of the Art

The first stage for attaining the general objective of this dissertation is to create a model which allows to formalize the elements involved in the controllability perspective of the use of assets. Since control is evaluated in terms of the compliance with some organizational policies, it was first needed to analyze the current

approaches to represent requirements on services. The works presented below were analyzed having in mind my target problem and the sought objective of this proposition, instead of their limitations regarding their own motivations. That is to say, current approaches are analyzed to determine whether they are appropriate or adaptable to this subject. The criteria considered for the review of the state of the art were the kind of requirements able to be represented as well as the formalization of the proposed approaches.

Workflows

Workflows, as defined in Section 2, govern the relation between different business processes either in an intra- or inter-organization way. In particular, even if services can be reused inside the organization, I focus on the analysis of Inter-Organizational Workflows (IOWF), where several independent partners (organizations) participate in the provision of a service. In [18] is presented an approach aiming to align the IOWF with the flexibility of SOA. Although the proposed approach is based on well-defined workflow architectures able to represent capacity sharing, chained execution, subcontracting, loosely coupled workflows and case transfer, only functional and interactional criteria of the services are considered to add or remove services in the workflow composition. In [3], those same criteria are added to non-functional properties to define compositions. Here, *features* encompass functional and non-functional requirements, which, when disabled or enabled, modify the service composition. Concretely, the authors of this work associate those non-functional requirements with QoS properties, namely a set of runtime attributes (availability, response time and execution time).

In general, the representation of service requirements in inter-organizational workflows has the advantage of being supported by well-studied formalisms, notably, Petri nets. Approaches of workflows for representing composed services and inter-organizational collaboration (or even competition) commonly focus on the dynamic adaptation of the composition, the separation of responsibilities aiming at the autonomy of the partners involved in the IOWF and the representation of business interactions as workflow patterns. Regarding requirements for the interaction, current approaches [112] [110] [77] [49][13] mainly cover functional issues, while the few tackling non-functional requirements consider reputation, performance and security properties such as availability and authentication.

Certificates

The works presented in [6] [22] [8] [9] tackle a problem similar to ours. Indeed, the authors of these works highlight the lack of trust and transparency in services, which are attributed to the impossibility to inspect applications and data. So, a complete framework based on certificates is proposed. An external certification authority, in charge of the certification process, takes evidence collected from tests (pre-deployment) and monitoring in runtime (in-production) to deliver a certificate to guarantee that a single or composed service complies with non-functional requirements in terms of security. Moreover, evidences can be used together with functional requirements to select and compose services. The

main advantage of this approach is the use of machine-readable evidences as well as to propose a formal model of both the certification process and the service. However, security is the only non-functional requirement considered in this approach. Besides, representing business requirements involves the agreement with a common vocabulary to express the business rules, but the certification framework lacks semantics which restrain the adding of business rules to that framework. Finally, the evaluation of evidences is done in real-time which allows to revoke or renew a certificate; however in most cases business rules do not describe observable attributes but more coarse-grain activities which complicates their monitoring in real-time. Consequently, modifying the certificate-based framework to support business aspects requires the adaptation of the core of its process, which is not a suitable approach. Notwithstanding, the principle behind certificates is really close to the controllability on assets, notably in terms of the use of evidences, which also gives interesting insights such as the selection of a service based on hierarchical attributes associated to its abstract properties.

Machine-readable SLA

As defined in Section 2.3, a SLA is a document containing the terms which govern the service provision. In general terms, such a document can be written in natural language or in machine-readable language; in this Section I will focus on machine-readable languages since they facilitate the automation of processes such as service discovery and verification of compliance with the service terms.

One of the first approaches found in the literature to define SLA in the service provision is WSLA [75] which is based on the specification of templates and XML schemas to define and monitor QoS terms. Although the WSLA specification may include non-technical aspects related to business terms, more concretely, monetary ones, it mainly supports resource metrics, composite metrics and SLA parameters. In addition, this approach allows to carry out actions depending on the results of monitoring process (violations of SLO[1]).

WS-Agreement [5], like WSLA, bases the agreement specification on XML schemas. WS-Agreement is composed of three building blocks: a schema to specify the agreement, a schema to specify the template, and a set of port types and operations to manage the agreement lifecycle, which includes creation, negotiation, expiration and monitoring. Besides the QoS runtime, it is possible to represent business requirements about penalties and rewards, as well as to associate business values (cost and importance) to the SLO. The main drawbacks of WS-Agreement are its lack of a formal semantic and its limited expressiveness to support more complex business requirements.

SLA* [64] is another work that proposes machine-readable agreements. Unlike WSLA and WS-Agreement, this approach is not tied to any language. It supports general definition of services (not only web services) through the proposition of a domain and language-independent definition of the agreement. In this approach, instead of defining constraints as a relation between two elements (parameter-value), it is defined as a variable bound by a domain, which gives a more abstract expressiveness. Due to the fact that the approach focuses on the syntax, the authors of this work present as a major limitation its lack of semantic, in particular for the definition of action post-conditions.

[1]Also referred as Service Level Target in the ITILv3 specification

Unlike the three previous works, WSLA+ [82] supports the specification of SLA in inter-organizational scenarios through the incorporation of multiple-parties agreements. SLAng [69] is another approach addressing the inter-organizational service provision. In particular, the target service provision model is composed of three tiers: applications, middleware, and the underlying resources (network, storage) where each component of the tier is provided by a different organization. Here, it is supposed that each partner offers its own SLA specifying technical QoS (performance, availability, reliability,...), which are negotiated to create a composed agreement. Two kind of SLA are proposed: horizontal SLA, defined as a contract between different parties providing the same kind of service, and vertical SLA, which regulates the support parties derived from their underlying infrastructure.

WSOL [108][107] is proposed to manage single and composed web services. In this approach, the service's offering plays the role of a SLA or service contract[2] consisting on a formal representation of a single class of service, together with its constraints and management statements. However, as most of the SLA approaches, it focuses on the syntax, while the semantic is left to define a vocabulary of metrics and measurement units of the QoS. Finally, in the monitoring process, violations are recorded in a log and used to notify an accounting party, defined in the management statement, to adapt its behavior.

A variation of the traditional statement-based terms is proposed in [88]. In this work it is stated that an agreement should make the service rules explicit in a formal, machine-readable and interchangeable way. Therefore, they propose the representation of service terms in the form of rules by using the RuleML language. The main advantage of rules is their intuitive representation and their underlying formalism based on LP proof theory. This approach, named RBSLA (Rule-Based Service Level Agreement), adds constructors for event conditions, state changes, external data integration, deontic norms (obligation, permission and prohibition) and temporal reasoning. Once again, it focuses on real-time features, in particular, availability. So, it does not fully exploit the expressiveness of LP for representing business rules.

In conclusion, current approaches to represent requirements about the service provision are classified in functional and non-functional. Since I am interested in representing the expected use of the shared resources, non-functional requirements were analyzed in depth. Such analysis lead us to conclude that current non-functional requirements focus on technical and runtime features of the service, mainly targeting security and performance properties. Moreover, most of them focus on the syntax and not the semantic, which is an important aspect for the understanding of the contractual commitments and their negotiation. As stated in [63], current SLAs do not sufficiently fulfil the requirements on business "since they are thought with other technical objectives" which are implicitly confirmed in [87]. Moreover, the adaptation of current approaches to support controllability is not suitable since they aim at a runtime monitoring process to prove compliance with the agreement terms; business requirements, on the contrary, are not always observable or measurable. Conversely, for the definition of controllability, I need a machine-readable representation which holds coarse-grained activities as well as any resource over which the organization wants to keep control. Additionally, a clear semantics for the definition of those elements

[2] Both terms are used interchangeably in that work.

is needed to tackle the use of business specialized vocabularies. Intuitively, an alternative method to the runtime monitoring should be implemented to automatically or semi-automatically verify the compliance with the controllability requirements. In spite of this, the limitations and strengths of current works give insights to tackle control on assets such as the use of evidences as proposed by certificates, and the use of rule-based representation and formalisms as offered in some SLA approaches. Table 3.1 summarizes current works to represent service requirements by presenting their main limitations and strengths from the point of view of controllability.

3.3 Semantic Contracts

From the literature review, I have identified that aspects such as how stakeholders use shared assets or how they behave during the service provision are high-level business issues which, to the best of my knowledge, have not received the same attention as functional, interactional, performance or security issues. Thus, it has not been completely addressed in the current approaches of service requirements representations. In order to formally define the business control on the use of assets, I will consider the following postulates:

Postulate 1: *An agreement on the vocabulary is required in order to avoid any misunderstanding about the expected behavior of partners.* Due to the fact that the expected use of assets is expressed by imposing conditions to an external partner or by making claims about his own behavior, an agreement on the business terminology employed by each partner is needed.

Postulate 2*: *An alternative approach to runtime monitoring should be implemented to prove compliance with business rules. Due to the business and technological independence of partners, as well as legal issues, every single action carried out by the external partner using the shared resource cannot be traced.

Postulate 3*: *The proposed approach should be able to verify coarse and non observable business activities.

In order to clarify the terms under which the collaborative relationship between clients and providers is based regarding the use of assets, some strategy should be implemented in the middle of such a relationship to govern the behavior of partners taking in mind the aforementioned postulates. Thus, it should be clearly defined what each partner *can, can not, should, should not, must and must not* do with assets in the context of the service provision. This way of "regulation" is not opposed to the SOA principle of transparency and loosely-coupling, since neither the service provider nor client reveals its internal processes, but some business policies are agreed between them for establishing the terms of their interaction in order to prevent organizational damages at both sides of the service provision due to the asset misuses.

Taking into account the controllability needs and the limitations of current approaches, I propose the implementation of *service contracts* to represent non-functional business requirements about the use of assets. As stated in [87],

Table 3.1: Approaches for Representation of Service Requirements

Approach	Strengths	Limitation
Workflows [3][18][112] [110] [77] [49][13]	• Based on formal models. • Target the autonomy of each partner.	• Functional requirements. • Runtime service attributes (performance) for non-functional requirements.
Certificates [6] [22] [8] [9]	• Use of evidences for guarantee service properties. • Machine-readable evidences. • Formal models of services and certificate process. • Hierarchy of security properties based on their attributes.	• Only security as non-functional requirement. • Lack of semantics • Evidences are collected and evaluated in runtime (in the in-production approach).
SLA/BSLA/RSLA [82] [75] [5] [64] [69] [63] [107] [108] [88]	• Machine-readable. • Automatic monitoring. • Service terms include functional and non-functional (monetary terms, performance and availability) requirements. • Highly adaptable. [5] • Business values associated to SLOs.	• Lack of formal semantic (except for [64]). • Limited expressiveness for business requirements. • Monitoring of agreement compliance only for runtime attributes. • SLOs defined as a parameter-value (except for [88]).

contracts differ from SLA in the way they are able to represent more general terms involving legal and organizational commitments. Thus, service contracts have a rich expressiveness compared to SLAs.

$$SLA \subseteq Contracts$$

I highlight the importance of explicitly defining what a service contract means in the frame of this proposition since this concept has been commonly used in an interchangeably way with terms such as SLA and WSDL [87]. I state therefore the difference between my vision of service contracts and the way in which they have been used in SLA and WSDL in terms of the elements that each one is able to represent and the organizational level in which they are defined. Thus, I consider two levels to which these terms could apply: business and technological one (Figure 3.1).

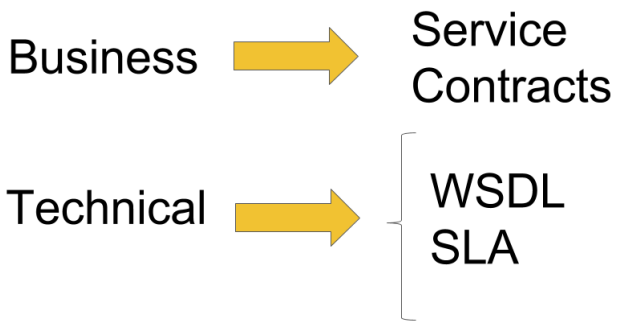


Figure 3.1: Contracts vs SLA

On the one hand, from the business point of view, organizations make use of the high expressiveness of the natural language to reflect in plain-text documents the aspects that protect them against the damages that an external organization may cause. Those documents, defined here as service contracts, include terms such as disclaimers, accounts, data use, contract modification, service termination, business requirements or legal aspects. These terms reflect the goals of the organization to protect itself against client dissatisfaction, customer loss, economical damages, loss of reputation and the like.

On the other hand, the policies that govern the relation between clients and providers at the technological level are focused on protecting the information system (web service) against security attacks and guarantee the performance of the web service. Such policies are specified in machine-readable documents (SLA and WSDL) in order to automatically verify their accomplishment. Even though SLA and WSDL reflect policies about the service provision, they respectively focus on non-functional and functional aspects of the interaction of the external partner with the web service, instead of reflect the terms that allow the organization to protect itself against damages resulted of the integration of an external partner into the business process.

Consequently, I state that service contracts are a wider concept reflecting the wishes and needs of both clients and providers related to the service provision, adversely a service agreement (SLA) is a narrow concept in which some

guarantees about stakeholders' interactions with the web service are reflected. However, even if the former represents richer terms useful for risk management, the latter has the advantage of having a machine-readable representation, which allows to automate processes such as negotiation and monitoring. Therefore, I aim in the first part of this thesis at the modeling of machine-readable contracts which will be able to represent non-functional business obligations and wishes regarding the use of assets. Moreover, such a representation is proposed as a semantic model which allows to formally define the components of a service contract.

When talking about machine-readable data, two approaches naturally emerge: semantic web and ontologies. The semantic web aims at the annotation of information with metadata useful to automate the processing of the data they describe. Nevertheless, this annotation faces the challenge of processing different data representation. So, ontologies overcome this problem by formally defining representations which facilitate the exchange of data and knowledge. As defined in "*Enabling Semantic Web Services: The Web Service Modeling Ontology*" [38], ontologies "formally define the structure and meaning of machine-readable metadata vocabularies". Ontologies use of formal representations to create models defining the semantic of data. Knowledge representation based on ontologies have been commonly used for:

- Capture, organize, reuse and share knowledge.
- Make inferences in a similar way to humans draw them.
- Represent knowledge by using formal models able to be read by a machine.

Although representing the world through models is a recurrent need in several domains such as chemistry, physics, informatics or artificial intelligence, the construction of a model is not a trivial issue. Indeed, subjectivity in the model should be avoided and a consensus among the stakeholders of a given model is required which proves its compliance with the reality it intends to represent. In this dissertation, I propose the use of a semantic model of service contracts to represent the expectations and obligations on the use of organizational assets. This choice is justified by:

- Ontologies allows to fill the gap between the representation of business requirements and their machine-readable form.
- It allows each partner to describe its own rules regarding the use of assets while keeping a clear meaning on their semantics, thanks to a common understanding of their vocabulary.
- Semantics allows to describe the knowledge about the contractual relations within a formal representation. It allows a clear separation of formal definitions and the syntax.
- From this representation some reasoning can be done to draw conclusions about the service provision regarding the parties behavior and the use of assets.
- Due to the ability of ontologies to reuse the formalized knowledge, more complex representation can be integrated to my model in order to extend some definitions without affecting the overall representation.

Knowledge representation techniques based on ontologies have already been implemented in the web service definition at the Business-to-Business level to automate the integration of services, which has created new areas of research called *semantic web services* and *service-oriented semantic web*. However, unlike current approaches based on the annotation of web service functionalities, I aim at annotating services with contracts containing usage policies.

The Description Logic (DL) formalism is the core of the ontologies definition by allowing to add a formal semantic to the specification of concepts as well as offering mechanisms of inference. DL has higher expressiveness than the propositional logic, but less than the First Order Logic (FOL), which allows to tackle the decidability and computational time of their concrete representations. The vocabulary of DL is composed of:

- Concepts: they are objects or classes of objects belonging to the particular domain of knowledge to be represented. For instance, Contract, Provider, Client, Service.
- Roles: roles are relationships linking concepts.
- Individuals: they are concrete instances of concepts. It means elements belonging to a class. For instance, ABC Enterprise, ACME Company.
- Axioms: they are statements relating concepts and roles. For example, Provider <provides> Service.
- Constructors: they create complex descriptions by using restrictions over concepts and roles. For example, a Service <hasExactly> 1 Provider.

The semantic in DL is represented by its theoretic model based on interpretations $\mathcal{I} = (\Delta\mathcal{I}, \cdot\mathcal{I})$. Where $\Delta\mathcal{I}$ is a non-empty set representing the domain $\mathcal{I}$, and the interpretation function $\cdot\mathcal{I}$ maps every concept $\mathcal{A}$ and every role r to a subset $A^{\mathcal{I}}$ and to a binary relation $r^{\mathcal{I}}$ over $\Delta\mathcal{I}$, respectively. Each member of the DL family of languages differs from the others by its expressiveness. The simplest DL language is the $\mathcal{AL}$ to which some constructors are added to create more complex representations. A mnemonic letter is associated to each of this constructors:

- $\mathcal{N}$ for numeric restrictions
- $\mathcal{Q}$ for qualified numeric restrictions
- $\mathcal{F}$ for functional properties
- $\mathcal{O}$ for nominals (one-of)
- $\mathcal{H}$ for role hierarchies
- $\mathcal{I}$ for inverse role
- $\mathcal{E}$ for existential qualification
- $\mathcal{U}$ for concept union
- $\mathcal{C}$ for complex concept negation

- $\mathcal{R}$ for limited complex role inclusion axioms, reflexivity, irreflexivity and role disjointness
- ($\mathcal{D}$) for datatypes properties, and datatype values.

Such mnemonics are also used to define the name of the language[3]. For example, the expressive power of $\mathcal{SROIQ}(\mathcal{D}+)$ logic is able to formally represent the basic constructors of $\mathcal{AL}$ (concept intersection, universal restriction, existential quantification and complement) plus complex role inclusions axioms ($\mathcal{R}$), reflexivity, irreflexivity, role disjointness, nominals ($\mathcal{O}$), inverse role ($\mathcal{I}$), qualified numeric restrictions ($\mathcal{Q}$) and datatype properties ($\mathcal{D}+$). Appendix A.1 and A.2 respectively present the expressiveness of some of the most used DL-sublanguages together with their complexity, as it is reported in [119].

A knowledge base built over a DL formalism defines two main components: a TBox(T) and an ABox(A). The former contains the ontology terms (concepts and roles), and the latter contains assertions about those terms. Inference engines reasoning over these knowledge bases allow to carry out particular inference activities over each of these two components, notably:

- *Satisfiability and subsumption checking:* satisfiability guarantee that new concepts added to the knowledge base are not contradictory with the existing ones. Formally, satisfiability is guaranteed if the new concept has an interpretation that is a model of T (a model that satisfies all the axioms of T). Subsumption on the other hand, checks hierarchies of concepts. Formally, a concept C is less general that another concept D if each model of C of T is a subset of a model of D.
- *Consistency checking and instance retrieval:* An ABox is consistent if an interpretation $\mathcal{I}$ exists that is a model of both A and T. Additionally, DL reasoners could also retrieve individuals belonging to a given concept.
- *Query answering:* query answering is a relatively recent reasoning task, which justify the fact that it is not supported by all current reasoners. Query answering allows to create complex expressions, which are used to query the ABox in a similar way as it is done in traditional database management systems.

Taking into account the previous characteristics, a semantic model of contracts overcomes the issue that each organization uses a specialized vocabulary, and it takes the advantage of the DL formalism, namely reasoning, query, high expressiveness and decidability. Moreover, it allows a machine-readable representation of contracts.

Definition 1. *(Machine-readable Service Contract)*
A machine readable contract is a document agreed between service clients and providers, which is understandable for the machine and reflects the terms that govern their collaborative relation. Particularly, it represents the expected stakeholder's behavior regarding the service provision.

[3]There are two exceptions. $\mathcal{S}$ that stands for $\mathcal{ALC}$ and $\mathcal{EL}++$ that stands for $\mathcal{ELRO}$

In this approach, contracts explicitly represent policies governing the relation between clients and providers regarding the use of assets. As a first step, before the definition of policies, I aim to propose a model in order to formally define contracts by using the DL formalism. Ontologies are not used here with the purpose of creating a taxonomy, but a vocabulary to add a clear semantic to the contractual terms that will be used to describe the policy. For instance, if the term "contractual parties" is used in the policy, its semantic allows to know that it refers to "the client" and "the provider". It also has the advantage of avoiding the subjectivity in the interpretation of the policy by supporting the semantics of its vocabulary in a formal model.

Therefore, I consider service contracts as the specific domain of knowledge to be represented. Following, a model of the knowledge belonging to such domain is presented. In order to determine what specific subset of the DL formalism should be used, I first identified the needs of expressiveness according to the following methodology:

1. Contract base collection: to create a model as much generic and expressive as possible that effectively supports the representation of business needs, I build a contract base composed of two kinds of contracts. The first set corresponds to real documents written in plain English containing business terms about the commitments of each party about services provided in the cloud. A total of 12 service contracts were analysed each one belonging to a particular layer of the cloud stack, i.e. data/document storage, infrastructure, and e-commerce applications. The second set of contracts corresponds to more general service contracts, i.e., services which are neither provided by the cloud nor mediated by a web service. This second group allows to identify business policies associated to more general organizational assets. In the following sections, a contract of each one of those groups was selected to illustrate my contributions.

2. Contract tagging: in this step, each of the documents of the contract base was manually analysed and key elements according to a controlled English[4] were identified and tagged. The aim of this step is twofold. First, it allows to define the structure of service contracts. Secondly, it creates a conceptual definition of the elements represented in the contract. With the tagging process, a contract vocabulary consisting of concepts and definitions is built. More specifically, each statement of the contract was individually analysed in order to identify contract concepts, specific names (individuals), verbs expressing relations between contract concepts (roles), and basic language symbols such as quantifiers, connectors, modals[5], and quantifications.

3. Formal representations: once the contractual terms were identified, their semantics is defined by using the logic formalism of the DL. In particular, the needs of expressiveness were matched with the constructors proposed by the different subsets of DL.

With those three steps, it is possible to create formal definitions of both the contractual concepts and the terms associated to the control of assets. However,

[4] A controlled English is assumed to reduce ambiguity.

[5] Modals help us to identify the nature of the contractual rules (obligations, recommendations, prohibitions, permissions).

a final step is added to the methodology to validate both the expressiveness of the chosen logic representation and the business requirements written in plain English so they can be mapped into a machine-readable representation.

4. Machine-readable representation: in this step the abstract model represented by the DL formalism is translated into a concrete ontological representation in XML by using languages and engines supporting the logic representation and inference.

By following this methodology, the $\mathcal{SROIQ}(\mathcal{D})$ formalism was used, since it fulfills the required level of expressiveness (such expressiveness will be described in detail in the following sections), while keeping decidability. However, despite the high expressiveness of this formalism, I strive to make clear that the present proposition of machine-readable service contract has been developed under the basis of two restrictions:

Postulate 4. *It was assumed that contracts are signed only between two parties, one acting as a provider and the other one acting as a client.* Consequently, federate contracts (contracts agreed among all the partners involved in a workflow) are not considered in this approach. Instead, the service provision chain is broke down in several client-provider relations, and the controllability terms (including third parties' obligations) are tackled by means of propagation of policies. As a consequence, in case of orchestrated services, several contracts govern the interaction of partners at the same time.

Postulate 5. *As a consequence of the previous postulate, the negotiation process is out of the scope of this thesis.* So, this proposition assumes that the policy governing the service provision has been already agreed between the given client and provider.

3.4 A Formal Model of Semantic Contracts

In order to effectively use contracts (and their associated metadata) to gain control in assets and in case of claims, I aim to formally represent as closely as possible, the elements and structure traditionally found in plain text service contracts and in organizational policies, while keeping a high degree of generality that allows this model to be used in a wide range of specific-domain applications.

Contracts are structurally composed of three building blocks, namely: a contract header, a contractual policy and signatures (Figure 3.2). Usually, a special section dedicated to the definition of terms is also found in plain-text documents, however that section was not explicitly defined in this proposition since the semantics of the contractual terms is given by the formal model.

Following, formal descriptions of the elements (concepts) identified in each of these components are proposed. For each element, a set of axioms is defined to establish its relations with other elements of the model. However, for the sake of clarity each axiom is presented and analyzed individually. Therefore, the complete definition of the concept should be understood as the conjunction of those individuals axioms. Even if the representation of roles is highly important in semantics (since they allow to describe meanings by relating several

concepts) for the sake of understandability, only a subset of the actual implemented relations is presented. In particular, if $r_1, {r_1}^- \in \mathcal{R}^{\mathcal{I}}$ and semantically $r_1 \equiv ({r_1}^-)^-$, only r_1 will be described in this section. In addition, cardinality restrictions are only explicitly presented when necessary. The complete set of relations and their cardinalities are presented in Section 3.5.

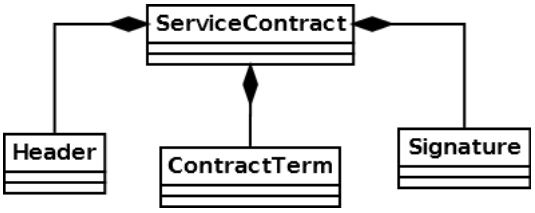


Figure 3.2: Contract Structure

3.4.1 Contract Header

The header is an introductory section which identifies the provided service, the role played for each party involved, as well as contract properties such as the version, effective date and publication date. In the header, I am interested in describing the semantics of the service, the service's actors and properties of the contract. An example of a contract header is presented in Figure 3.3.

SERVICE CONTRACT

Publication Date: August 01st 2015
Effective Date: August 12th 2015

This service contract ("Contract") is made on the August 01st 2015 at 08:10 between

1. PPMS ("Service Provider") established in France at 12 Rue Blue, 4324, Aquitaine.
2. Oil&Gas Company ("Service Client"), an oil company located in 1234 NE 12 AVE, Miami, FL 33135, USA

This contract governs the management of oil projects ("Service") by PPMS. The "Service Client" wishes to be provided with the "Service" by the "Service Provider"; and the "Service Provider" agrees to provide the "Service" to the "Service Client" on the terms and conditions of this contract.

....

Figure 3.3: Contract Header Example

Actor

In order to bind each service stakeholder through contractual obligations, it is required to identify and classify the actors involved in the service. In this approach, the service provider, service clients, third parties and signatory parties

are considered as contractual actors with associated roles from the point of view of controllability. Particularly, providers and clients are used in the same way as defined in the service oriented architectures. That is to say, the provider is seen as an active actor supplying a service requested by the client.

Definition 2. *(Actor)*
An actor is any active entity involved in the provision of the service, and who is directly referred in the contract either as a specific individual or as a category.

I use Postulate 4 and Definitions 1 and 2 to formalize the following semantic expressions:

$$Contract \supseteq \{c \mid \forall (c, s) \in governs \rightarrow s \in Service\} \tag{3.1}$$

$$Contract \supseteq \{c \mid \forall (c, a) \in involvesParty \rightarrow a \in Actor\} \tag{3.2}$$

$$(Provider \cap Client) \cap ThirdParty = \varnothing \tag{3.3}$$

In the modeling of the semantics of a service contract, it was assumed that the contract governs one single service. It implies the definition of at least two contractual parties describing who requests the service (client) and who provides it (provider), the role of each actor in the contractual relation is defined by means of the property `involvesParty`:

The Eq. 3.3 holds since `Provider`, `Client` and `ThirdParty` are all non-empty sets and the intersection of the empty set with any set is the empty set. Consequently, in the proposed model, no actor can have two contractual roles at the same time, i.e. it is not possible to be defined as a provider and as a client for the same service. I strive to clarify two important issues regarding this restriction. First, note that Eq. 3.3 does not restrict all actors' subclasses to be disjoint between them, only the subset corresponding to the contractual roles (contract parties). So, as it will be shown below, it will be perfectly possible from the point of view of controllability, for an actor to detain some obligations, but at the same time, to impose some other obligations, which makes him grant two roles. That scenario, however, will be only possible with controllability roles and not with contractual roles. Secondly, in spite of that restriction, it is common to find scenarios in which an organization *Org1* requests a service to an organization *Org2*, which, in turn, requests part of the service to an organization *Org3*. Those cases are not only usual but also make evident that a given organization could play the role of both provider and client (see Figure 3.4). Considering that my proposition does not tackle federate contracts, I assume that in those cases two contracts are established C_1 and C_2, the first one between *Org1* and *Org2*, and the second one between *Org2* and *Org3*, where in C_1, some administrative rules are created specifying the switching of roles, in case of propagation of rules from C_1 to C_2.

$$Contract \supseteq \{c \mid \#\{p \mid (c, p) \in involvesParty \wedge p \in Provider\} = 1\} \tag{3.4}$$

$$Contract \supseteq \{c \mid \#\{cl \mid (c, cl) \in involvesParty \wedge cl \in Client\} = 1\} \tag{3.5}$$

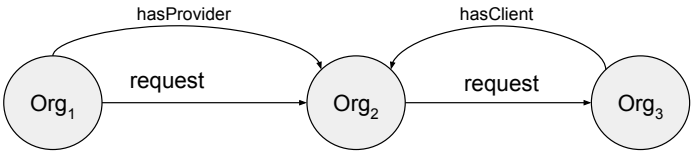


Figure 3.4: Multiples Roles for an Actor

Besides clients and providers, third parties are also part of the definition of actors. They include certification authorities, subcontractors of the provider, auditors and suppliers. From the legislation point of view, most laws of contract around the world state that only contractual parties, i.e. service clients and providers, can enforce a contract. However, specific ordinances, such as those in Hong Kong and in the UK, give some benefits and rights to third parties[6]. As prescribed by law, in case a contractual party confers benefits or rights to a third party, this latter should be specifically identified in the contract by its name or as a member of a class or as answering to a particular description. Other kinds of contracts aims to explicitly establish the nature of their collaborative relation. For example, an excerpt of the terms of service of Dropbox[7] establishes "[...] You give us permission to do those things, and this permission extends to our affiliates and trusted third parties we work with". Consequently, third parties are identified in the tagging process as concepts whose semantics need to be defined in terms of their relation with the service, and more importantly, in terms of their relation with assets.

$$ThirdParty \subseteq (Actor \wedge \neg Client \wedge \neg Provider \wedge \neg SignatoryParty) \quad (3.6)$$

Finally in the definition of actors, a signatory party is an entity legally acting on behalf of the client/provider to attest that the party agrees with the contract. So, even if the client and provider are both bound parties, who actually signs a contract is not the organization as a whole but an individual actor with the authority to represent the organization. Note that in the present model, the signatory party is neither defined as a subclass of provider nor as a client due to its semantic consequences for the reasoning. Indeed, subclasses are subject to the same inferences as superclasses, which may result in some imprecise semantics in the definition of the role of the signatory party. To illustrate this, let us consider the semantic consequences of a subsumption axiom regarding the property `provides` which defines the actor who provides the service. If the signatory party were defined as a subclass of provider, the property `provides` will be also valid for the signatory party, which will result in a semantically incorrect meaning regarding the provision of the service. That is to say:

(i) A provider provides a service
(ii) A signature party is a provider //With a subsumption axiom
$\Rightarrow$ *A signatory party provides a service* //Not true!!

[6] However, they clarify that third parties cannot have obligations imposed by the contract.
[7] `https://www.dropbox.com/privacy?view_en#terms`

What is more, due to the fact that the contract is restricted to having one single provider (resp. client), the reasoner, in order to hold Eq. 3.4, is able to infer that the signatory party and the provider (resp. client) are the same actor, which once again is an undesirable inference.

$$Client \supseteq \{cl \mid \forall(cl, s) \in requests \rightarrow s \in Service\} \tag{3.7}$$

$$Provider \supseteq \{p \mid \forall(p, s) \in provides \rightarrow s \in Service\} \tag{3.8}$$

$$SignatoryParty \supseteq \{sp \mid \#\{cl \mid (sp, cl) \in identifies \wedge cl \in Client\} = 1\} \tag{3.9}$$

$$SignatoryParty \supseteq \{sp \mid \#\{p \mid (sp, p) \in identifies \wedge p \in Provider\} = 1\} \tag{3.10}$$

Service

One semantically difference between this model and most of the current SLA is the definition of service. Indeed, as I aim at the formalization of business rules, my definition of service is consistent with that business perspective. In this approach, services are considered in their broad sense, as any process offered by a service provider to a service client, by keeping the properties of transparency, flexibility, and business and technological independence between clients and providers. Thus, the result of the contractual relation between clients and providers is some tangible or intangible commodity, which represents the core of the business logic of the provider, as well as the final "product" of the contractual relation. Therefore, the definitions of both service and commodity are highly important since the conformity of the commodity with the client's expectation is part of the successful assessment of the quality of the service.

Definition 3. *(Service)*
A service is the object of the contractual relation in which the provider supplies some tangible or intangible commodity requested by a client.

The definition of the service, in the context of a contractual relation, is formalized as:

$$Service \supseteq \{s \mid \forall(s, com) \in \text{produces} \rightarrow com \in \text{Commodity}\} \tag{3.11}$$

$$Service \supseteq \{s \mid \exists(s, attr) \in \text{hasAttribute} \wedge attr \in \text{ServAttr}\} \tag{3.12}$$

$$Commodity \supseteq \{com \mid \exists(com, attr) \in \text{hasAttribute} \wedge attr \in \text{CommAttr}\} \tag{3.13}$$

Where `ServAttr` and `CommAttr` are concepts grouping the set of features defining the attributes of the service and commodity, respectively, for a particular organization. For example, for an organization offering the service of selling home appliances online, a commodity can be a refrigerator, while an attribute of the refrigerator is the color. In general, if those attributes are defined as part of the contractual terms, their compliance becomes a key factor in the fulfillment of the expectations of the contractual parties regarding the provision of the service, which is reflected in the assessment of both the service and its involved parties.

Attributes

Attributes are highly important because they are an essential part of the definition of the contract concepts by describing the properties associated to them. Up to now, attributes are been abstractly defined by means of the relation `hasAttribute`. Those concepts aim to capture the fact that a particular element (contracts' concepts) has some features, to which some values are associated. This knowledge can be easily represented in the FOL as $hasAttribute($ $< concept >, < attrName >, < attrValue >)$. However, the theoretic model of the description logic assumes $R = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. It means, a role is assigned to a pair of individuals in the form of 2-ary operations. Therefore, the previous 3-ary relation is reformulated to express attributes as:

$$hasValue(a, < attrValue >) \tag{3.14}$$

where {a ∈ hasAttribute(<concept>, <attrName>)}, $\{concept \in \Delta^{\mathcal{I}}\}$ and $\{attrName \in Attribute\}$. Consequently,

$$\{\exists e \mid \forall att(e, att) \in hasAttribute \rightarrow att \in Attribute\} \tag{3.15}$$

Two kinds of attributes are considered according to the knowledge represented by the *attrName* (which, in turn, determines the value of the *attrValue*), namely: quality attributes and relational attributes. The former assigns a data-typed value to an attribute, while the latter defines the value of an attribute as being an individual belonging to the domain of interpretation. Relational attributes are particularly interesting since they overcome the limitation of most of the existing approaches which only represent attributes as a parameter-data value relation.

$$\text{Attribute} \subseteq \{\text{RelationalAttribute} \cup \text{QualityAttribute}\} \tag{3.16}$$

$$\text{RelationalAttribute} \cap \text{QualityAttribute} = \emptyset \tag{3.17}$$

$$\text{QualityAttribute} \subseteq \text{Attribute} \wedge \neg\text{RelationalAttribute} \tag{3.18}$$

Eq. 3.16 and Eq. 3.17 reflect the fact that the set of attributes is formed by some quality attributes and some relational attributes, when a particular individual can not belong to both subsets as depicted in Figure 3.5.

Although according to the DL vocabulary, properties are, in general, expressed as roles, it is not a misrepresentation in my model to describe them as individuals belonging to a class. From the point of view of the contractual policies, attributes are not only metadata associated to a concept, but as it was previously stated, some business restrictions can also be expressed in terms of those attributes. For example, a rule about the state of some other rule or about some physical feature of a commodity. I argue that if the behavior of the processors need to be controlled regarding those attributes, then they become organizational assets themselves. Therefore, this knowledge can be captured by considering the attributes as a concept. It means, by using the predicate-based notation, I move from a representation like *hasVersion(contract, 1.0)* in

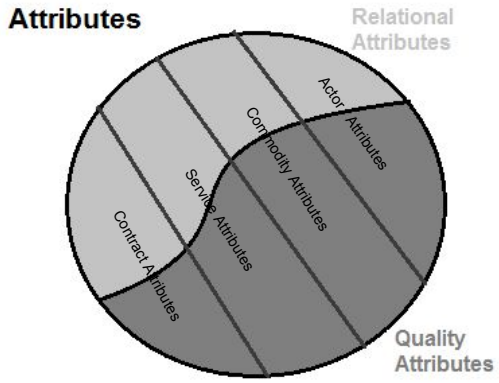


Figure 3.5: Subset of Attributes

which *Version* is expressed as a relation to a representation like [$\mathcal{A} \doteq$ *hasAttribute(contract, version)*; hasValue($\mathcal{A}$, 1.0)]. It has the advantage of allowing to associate more complex descriptions to the properties themselves, and facilitates the attribute-based query to the knowledge base.

$$\begin{aligned} hasLiteralValue &\subseteq hasValue \\ hasClassValue &\subseteq hasValue. \end{aligned} \tag{3.19}$$

$$RelationalAttribute \subseteq \{ra \mid \forall(ra, val) \in \text{hasClassValue} \rightarrow val \in \Delta^I\} \tag{3.20}$$

$$\{\forall x|(x, y) \in \text{hasLiteralValue} \rightarrow x \in \text{QualityAttribute}\} \tag{3.21}$$

$$ActorAttribute \subseteq Attribute \tag{3.22}$$

$$Actor \subseteq \{a \mid \forall(a, at) \in hasAttribute \rightarrow at \in ActorAttribute\} \tag{3.23}$$

In order to facilitate the visualization of the relations among the elements aforementioned described, in Figure 3.6 an schematic view is presented. Note that since, Eq. 3.22 and Eq. 3.23 hold for `Actor`, `Commodity`, `Service`, and `Contract`, the concepts `CommodityAttribute`, `ServiceAttribute`, `ActorAttribute` and `ContractAttribute` were omitted.

3.4.2 Contract Terms

Although, the definition of policies is explained in detail in Chapter 4, I will present some meta-information useful for the definition of the contractual terms.

Asset

Taking in mind that the goal of the targeted policies is to explicitly define rules governing the use of organizational resources, the definition of what an asset means and the way in which it relates with the components of the service contract becomes a key aspect of this model.

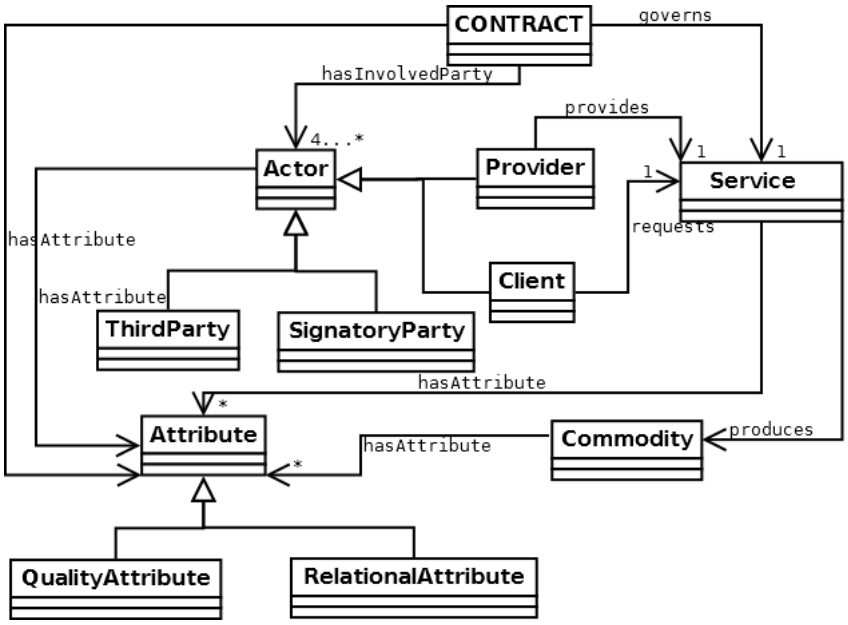


Figure 3.6: Semantic of the Contract Header Terms

Definition 4. *(Asset)*
In the contract model, an asset is any resource having some value for an organization and on which some rules regarding its use are established as part of contractual terms.

According to the ISO/IEC 27005 [56], organizational assets are divided into two groups, namely, primary assets and supporting assets. In general, the business operations, processes, activities and information sensitive for the organization are considered as primary assets. It means, resources which could compromise the mission of the organization. Whilst to the second group belongs those organizational elements that may compromise the primary assets if their vulnerabilities are exploited. This includes the hardware, software, network, personnel, site and the organization's structure. This approach relies on the fact that a subset of those assets is shared or delegated during the service provision, it means they totally or partially left the organization and become part of the business operation of an external processor. However, even if the asset is externally used, the organization initially detaining some rights on them requires (and expects) that they be used in a certain way in order to avoid any organizational damage due to misuses.

Concretely, assets are modeled in terms of their relations with the actors during the service provision. Those relations specify the usage in terms of any possible interaction of the actor with the asset. The relations `isControlledBy` and `isUsedBy` relate the actor who restricts the activities that are done with the asset (Eq. 3.24), and the external actor who, according to the contractual terms, is bound to comply with those usage restrictions (Eq. 3.25).

Taking in mind this definition of assets, it becomes clear that the aimed policies cover a wider range of organizational resources than only files or data,

as the traditional security policies.

$$Asset \supseteq \{as \mid \forall(as, ct) \in \text{isControlledBy} \rightarrow ct \in \text{Controller}\} \tag{3.24}$$

$$Asset \supseteq \{as \mid \forall(as, pr) \in \text{isUsedBy} \rightarrow pr \in \text{Processor}\} \tag{3.25}$$

$$Asset \subseteq \{as \mid \exists(as, x) \in \text{employs} \wedge x \in \Delta^I\} \tag{3.26}$$

The role `employs` is very powerful since it represents any interaction between two organizational resources during the provision of the service, which allows to create chains of dependency among resources. For example, a provider who interacts with a third party, or an activity which `employs` one or several assets (see Eq. 3.37).

Data and Metadata as an Organizational Asset

The role played by the information systems within an organization has led them to being considered as a key aspect to accomplish business goals. Moreover, the data they collect and process has received even more attention as it generates competitive advantages and misuses may compromise the organization, which justifies the efforts to preserve and secure it. Data can be of any type (business, strategy, operational or personal information) and comes from different sources. In the context of B2B, data travels from one organization to another, making difficult to control its use, and most crucially, the automatic tracing of fine-grained actions about the use of the information goes against the SOA principle of transparency and technological independence; without mentioning that it may be, at least, illegal.

The security of information (framed in a perspective in which the loss of control in its use is associated to risks that could generate organizational damages) is not currently taken into account in the design of the security policies in SOA. In the European Union, the Data Protection Directive 95/46/EC [34] regulates the sharing, collection and processing of data, when the data represents personal information[8]. However, for the time being, more technological efforts are needed to be implemented in the SOA architecture to prove the external partner's compliance with such regulations.

My model of contract is able to support the definition of restrictions that apply to the usage of the data, which is by definition a particular case of an organizational asset. Similarly, automatically collected data, and other kinds of metadata, such as IP addresses, behavior history, location of the contractual parties (or third parties) or logs are also subject to restrictions. Currently, metadata is automatically collected during the provision of services, sometimes without the knowledge of users, without their consent or with an unclear understanding about their intended use. In this approach, on the contrary, it is proposed that the collection of any data/metadata be explicitly reflected in a machine-readable form with clear restrictions about what activities are permitted to be done with that data and those forbidden. Such restrictions are not focused on fine-grained actions such as read or write - which could be easily

[8] It is defined as anything that can identify an individual.

defined by means of traditional access control policies - but on more high-level business aspects linked to risk management such as disclosure of data, the purposes of the data collection and guarantees about the activities carried out with (for instance, the guarantee that data is deleted after a certain period of time).

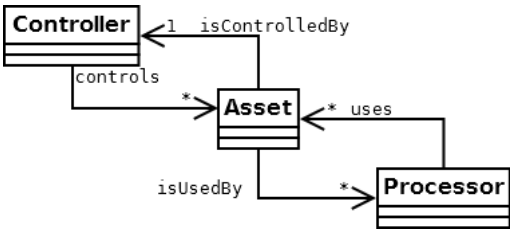


Figure 3.7: Semantic of the Organizational Asset

Controller and Processor

In order to represent the relation between assets and their stakeholders, the concepts controller and processor are modeled. Those terms represent the fact that even if assets are used by different actors, they "belong" to one entity who grants some rights over them to an external actor for a particular purpose. In [34], a controller is defined as any "natural or legal person, public authority, agency or other body which, alone or jointly with others, determines the purposes and means of the processing of personal data". Similarly, a "processor means a natural or legal person, public authority, agency or other body which processes personal data on behalf of the controller". In this model, those definitions are retaken but also extended to tackle the aforementioned category of assets.

Definition 5. *(Controller)*
A controller is a contractual actor establishing business rules about the use of the assets which are under the control of his organization.

Definition 6. *(Processor)*
A processor is a contractual actor enforced to respect the business rules established by the controller regarding the use of assets.

By signing the contract, the processor is responsible for complying with the contractual rules, i.e. with the use of the assets according to the controller's expectations and requirements. Note that while the concepts client and provider are attached to a semantics directly linked to the act of providing the service (which was previously defined as contractual actors, i.e. contractual parties), the controller and processor are both terms which make sense in the context of controllability (controllability actors). Moreover, in the policy as a whole, the provider (resp. client) could be a controller, a processor or both (Eq. 3.29 and 3.30), since the policy contains the set of all assets exchanged during the service provision either by the client or the provider. Then a provider can play the role of controller for an asset, but also plays the role of processor for another asset; but for a particular asset, a contractual party can not play both roles at

the same time. It means, an actor cannot be the controller and processor of an asset. This is formalized by the disjoint relation between `uses` and `controls` in Eq. 3.31.

$$Controller \supseteq \{ct \mid \forall(ct, as) \in \text{controls} \rightarrow as \in \text{Asset}\} \tag{3.27}$$

$$Processor \supseteq \{pr \mid \forall(pr, as) \in \text{uses} \rightarrow as \in \text{Asset}\} \tag{3.28}$$

$$Client \supseteq (\text{Processsor} \cup \text{Controller}) \tag{3.29}$$

$$Provider \supseteq (\text{Processsor} \cup \text{Controller}) \tag{3.30}$$

$$\text{controls} \cap \text{uses} = \emptyset \tag{3.31}$$

From the previous formalization, some knowledge such as {Processor employs some Asset}, which is not explicitly asserted in the knowledge base can be inferred by the reasoner because the Processor is inferred to be an actor. It leads to define a subsumption axiom between `uses` and `employs` as shown in Eq. 3.32. The difference between these two relations is that while the latter is a wider concept defining the dependence between two different concepts defined in the contract, the former entails a more specific relation which defines the expected use of an external partner with an organizational asset.

$$\forall(x, y) \in \text{uses} \rightarrow (x, y) \in \text{employs} \tag{3.32}$$

Regarding the definition of controllability rules, the concepts `Controller` and `Processor` are used to formally represent the relations establishing who sets the rule and who is the beneficiary of the rule defined in the contractual term. The former is justified by the fact that rules can be associated to claims that a party made about its own behavior or, on the contrary, they can be commitments that a party imposes to another one[9]. The latter models transitivity in the behaviors represented in rules. In other words, the effect of an expected behavior transits from one actor to other. For example, in an e-shopping domain where a contract governs the delivery service, a contractual obligation can be created by the client to enforce the provider about the notification to the final users in case of delay in the product delivery. In this example, the rule is created by the client, it obligates the provider, but the final user is the beneficiary.

$$Rule \supseteq \{r \mid \forall(r, a) \in \text{hasBeneficiary} \rightarrow a \in \text{Actor}\} \tag{3.33}$$

$$Rule \supseteq \{r \mid \forall(r, cp) \in \text{isSetBy} \rightarrow cp \in \text{ControllabilityActor}\} \tag{3.34}$$

$$ControllabilityActor \supseteq \text{Controller} \tag{3.35}$$

$$ControllabilityActor \supseteq \text{Processor} \tag{3.36}$$

[9] In case it is about self-claims, the evaluation of behaviors could lead to adapt their assertions about the service provision.

Business Activity

As it has been previously stated, in the contractual relations between clients and providers, organizations rarely reflect requirements in terms of fine-grained actions. On the contrary, more coarse-grained activities are described. Precisely, it is due to the lack of solutions allowing to represent and verify those requirements, that they have not been included in the machine-readable SLAs. In order to represent business terms in the controllability requirements, the concept `Business Activity` is defined.

Definition 7. *(Business Activity)*
A business activity represents a coarse-grained organizational operation in which the use of one or more assets is involved.

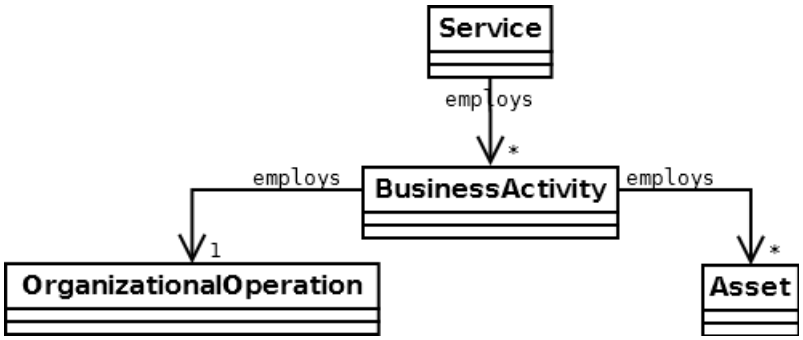


Figure 3.8: Business Activity

Therefore, a business activity represents the observable use of an asset within an organization. In other words, the operation represents any possible interaction with the asset, while the activity is the organizational term coined for that use. For instance, a notification is a business activity where the contact information represents the asset, and the act of sending is the operation. Note that in the model, a controllability actor (controller or processor) is linked to the asset, instead of the activity. Indeed, I am interested in modeling the fact that for the correct provision of the service, a set of activities needs to be carried out either by the provider or the client, and to do so, some assets are used. Business activities can therefore be seen as a concrete representation of the partners' behavior. Individuals belonging to the class `BusinessActivity` will be used in the definition of the policies governing the contractual relation in order to restrict the specific situations/conditions in which an actor can perform the activity.

$$Service \subseteq \{s \mid \exists(s, bo) \in \text{employs} \wedge bo \in \text{BusinessActivity}\} \tag{3.37}$$

$$BusinessActivity \subseteq \{ba \mid \exists(ba, op) \in \text{employs} \wedge op \in \text{OrganizationalOperation}\} \tag{3.38}$$

$$BusinessActivity \subseteq \{ba \mid \exists(ba, a) \in \text{employs} \wedge a \in \text{Asset}\} \tag{3.39}$$

Damage and Infringement

Similar to the definition of assets and its involved stakeholders, the identification of the organizational damages resulted from contract breaches is a core aspect of the controllability approach. Indeed, one motivation for the creation of the aimed contractual policies within an organization is to have guarantees that the vulnerabilities of its delegated/shared assets are not exploited by an external partner during the service provision, it is done by explicitly defining rules about the correct use of assets. In contracts, some terms usually refer to actions to take in case damage occurs. From the risk management perspective, the ISO guide 73:2009 [103] defines that damages are caused by events, which generate as consequences, negative effects for an organization. So, each type of damage may harm the organization in a different way according to the nature of the involved asset. For instance, damages to personal data are not going to cause the same organizational impact as damages to an equipment. In this model, the risk management perspective is retook to model the meaning of damages. However, unlike to the ISO standards in risk management, damages are not explicitly linked to the asset vulnerabilities. On one hand, the representation of vulnerabilities is out of the scope of a contract model; on the other hand, due to the fact that the machine-readable contract travels in the SOAP messages, it does not make sense that the vulnerabilities on assets be shared with external partners.

Definition 8. *(Damage)*
A damage is any negative consequence affecting an organization, caused by an intentionally or unintentionally event performed by an actor in the frame of the service provision.

Considering that a contract can be understood as a plan governing the behavior of actors, the events preventing damages to the organization should be explicitly represented in the policy as rules. Consequently, the undesired events which may cause damages are seen as a result of the non-compliance with one or more rules established in the policy, i.e, the non-compliance with some contractual term. In this proposed model, any deviation or non-compliance with the contract terms is defined as an infringement. I highlight that an infringement encompasses any non-compliance with a term regardless its nature. It means, it can refer to either the violation of a rule or the omission of a recommendation. Similarly, note that a damage is an organizational consequence of a breach in the contractual relation, such as loss of reputation, caused by the infringement of a contractual term. In general, damages are usually defined in contracts according to the category they belong, for instance, consequential damage or punitive damage, allow to hide the actual impact of a contractual breach for the affected organization.

Definition 9. *(Infringement)*
An infringement is an event representing the fail to fulfill some term established in the contract.

Particularly, as it will be presented in Chapter 5, I consider as infringement for a controllability rule, the lack of evidences, which support the actual behavior of the processor.

A service contract aims to clarify the terms of the relations between clients and providers by covering any issue that may affect the relation between contractual parties. Usually, those terms are identified by their nature, mainly, payment, guarantee, remedies, claims, liability, exclusion, contract cancellation/modification and legislation (Figure 3.9). In particular, guarantee terms are statements in which some contractual party ensures the compliance with some commitments regarding the provision of the service. On the other hand, remedies and claims specify rules which are not directly focused on the service provision but on the path to follow (behavior) when some contractual term, including service guarantees, are not respected.

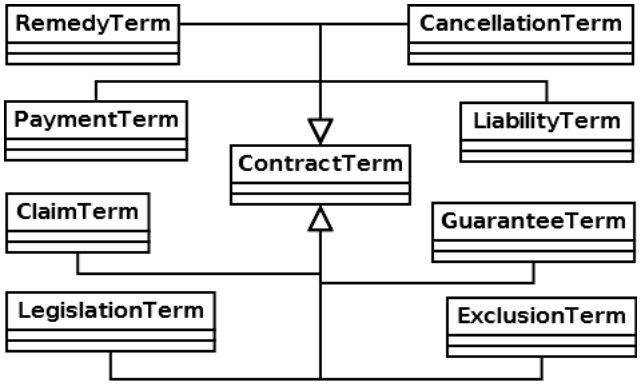


Figure 3.9: Taxonomy of the Contract Terms

Definition 10. *(Remedy)*
Remedies are defined as a compensation for failures to perform the rules governing the service provision or for a deviation of the terms agreed in the contract.

When some infringement occurs, compensations can be put in place to balance the effect of damages. The explicit inclusion of remedies in the terms agreed between a client and a provider is a current practice in SLAs. However, remedies are defined in monetary terms, usually such as service credits. From a business perspective, other kinds of material or immaterial remedies are also possible such as awards or preferential treatment. Examples are found in airline companies who, in case of overbooking, offer free hotel rooms, extra travels or some other amenities to passengers that voluntarily give their seats.

$$Remedy \supseteq \{re \mid \forall (re, inf) \in \text{compensates} \rightarrow inf \in \text{Infringement}\} \quad (3.40)$$

$$Infringement \subseteq \{i \mid \#\{t \mid (i,t) \in \text{infringes} \wedge \text{t} \in \text{ContractTerm}\} = 1\} \quad (3.41)$$

$$Infringement \subseteq \{i \mid \exists (i,a) \in \text{affects} \wedge a \in \text{Asset}\} \quad (3.42)$$

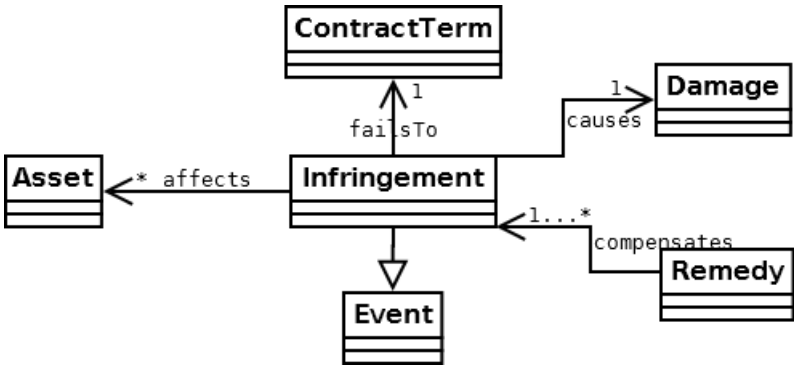


Figure 3.10: Semantics of Infringement

3.4.3 Signature

Digital signatures, analogously to their handwritten counterparts, are a strategy that allows a person to authenticate the content of a document[10]. Formally, they are based on mathematical algorithms that guarantee three main characteristics: authentication, integrity and non-repudiation.

Definition 11. *(Signature)*
Signatures represent the agreement between the service provider and the client to comply with the terms specified in the contract policy.

On the whole, the concrete contractual policy resulted from the translation of the proposed logic-based representation into some concrete syntax should be hashed in order to create its corresponding message digest. That message is encrypted with the private key of the contract party, resulting in a digital signature that is appended to the contract. This process is carried out for each party in order to validate commitments with regard to the compliance with the contractual policy. Unlike other agreement strategies such as implicit acceptance or click and accept, the advantage of using digital signatures in a machine-readable contract representation is their ability to detect contract modification by comparing the message digest obtained from hashing the unsigned contract and the message digest obtained after decrypting the signature of each party.

It is appropriate to point out that signatures are not dependent upon each other, but, both signatures are required. Signatures are not included in the message digest, and a signatory party is not obliged to wait until the other party signs the contract to append his/her signature. From the technical perspective, several approaches can be used to implement this issue. For example, if, in the pre-contract phase, a third party is in charge of negotiate the terms between the provider and the client, this entity could be also responsible for collecting the two instances of the contract signed by each party, for verifying that any change is not made and for re-distributing the signed contract. Another approach is very close to the process carried out by paper-based documents, where a party

[10] Digital signatures, as a particular case of the electronic signatures, have a legal value in most of the courts.

signs the contract and sends it to the counterpart for signature. Regardless the technique implemented, this approach considers that contracts have been already signed in some previous stage. The objective of this assumption is twofold. First, it allows us to support the semantics of the proposed concepts, for instance, a provider becomes a contractual party if he or she accepts the offer and is bound by the terms of the contract. Secondly, it allows to focus on the actual contribution of this research; let us recall that aspects such as the negotiation of the contract terms are out of the scope of this dissertation.

Due to the fact that web service technologies are highly standardized, the efforts made by the W3C community to secure (any) digital content through XML signatures are recalled. The "XML Signature Syntax and Processing" Recommendation [24] specifies the syntax for XML digital signatures together with its canonicalization via a schema XSD. To overcome its limitation of a formal semantics, semantic digital signatures are proposed in [48]. As a core part of a larger work aiming trust in web content, authors use semantic web technologies to define the meaning of digital signatures. So, they state that this representation allows the integration with other DL-based representations to support the context-aware meaning in signatures.

Although in this model, the decision of the syntactical parameters to use is left to the contract designer, four axioms are added to represent the role played by digital signatures in the definition of a contract. First, a contract needs to be signed by two contractual parties. Secondly, a signature included in the contract can only be of the individual having the legal right to represent the provider/client, which is formalized by the role `identifies` (as it was previously described, this relation allows to represent an individual by means of his/her category). The third axiom reflects the fact that the signatures of the contract authenticate its content, which gives it a legal value. Finally, the last axiom is used to model the fact that by signing the contract the contractual party `agrees` with the contractual policy. Eq. 3.46 is read as: if the contractual party C_1 `isIdentifiedBy` the by signatory party SP_1, and SP_1 `signs` the contract, then C_1 `agrees` the contract. The previous axiom holds because {`isIdentifiedBy`, `signs` $\prec$ `agrees`}.

$$Contract \supseteq \{c \mid \forall (c, sg) \in \text{hasSignature} \rightarrow sg \in \text{Signature}\} \tag{3.43}$$

$$Signature \supseteq \{sg \mid \exists (sg, sgp) \in \text{identifies} \wedge sgp \in \text{SignatoryParty}\} \tag{3.44}$$

$$Signature \supseteq \{sg \mid \forall (sg, c) \in \text{authenticates} \rightarrow c \in \text{Contract}\} \tag{3.45}$$

$$\text{isIdentifiedBy} \circ \text{signs} \subseteq agrees \tag{3.46}$$

Although for the sake of understandability only meaningful relations to express the meaning of concepts and make controllability inference have been presented in this section, Table 3.2 summarizes the vocabulary of the proposed model. It reflects the expressiveness of semantic service contracts based on the DL formalism. Figure 3.12 presents the semantic contract model with the concepts explained in this Chapter. In the following sections, the concrete syntax of the model is described in order to validate this representation and the proposed axioms. In Chapter 4, the modeling of the contractual terms is explained in depth, where some additional axioms are proposed to define the semantics of the targeted business rules.

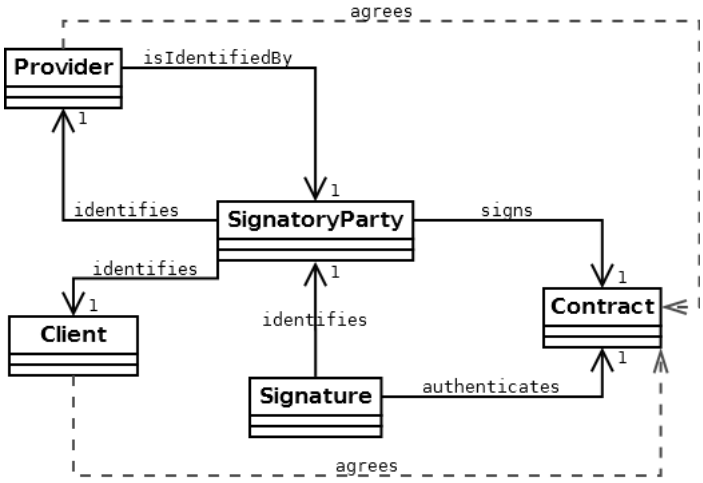


Figure 3.11: Semantic of the Contract Signatures

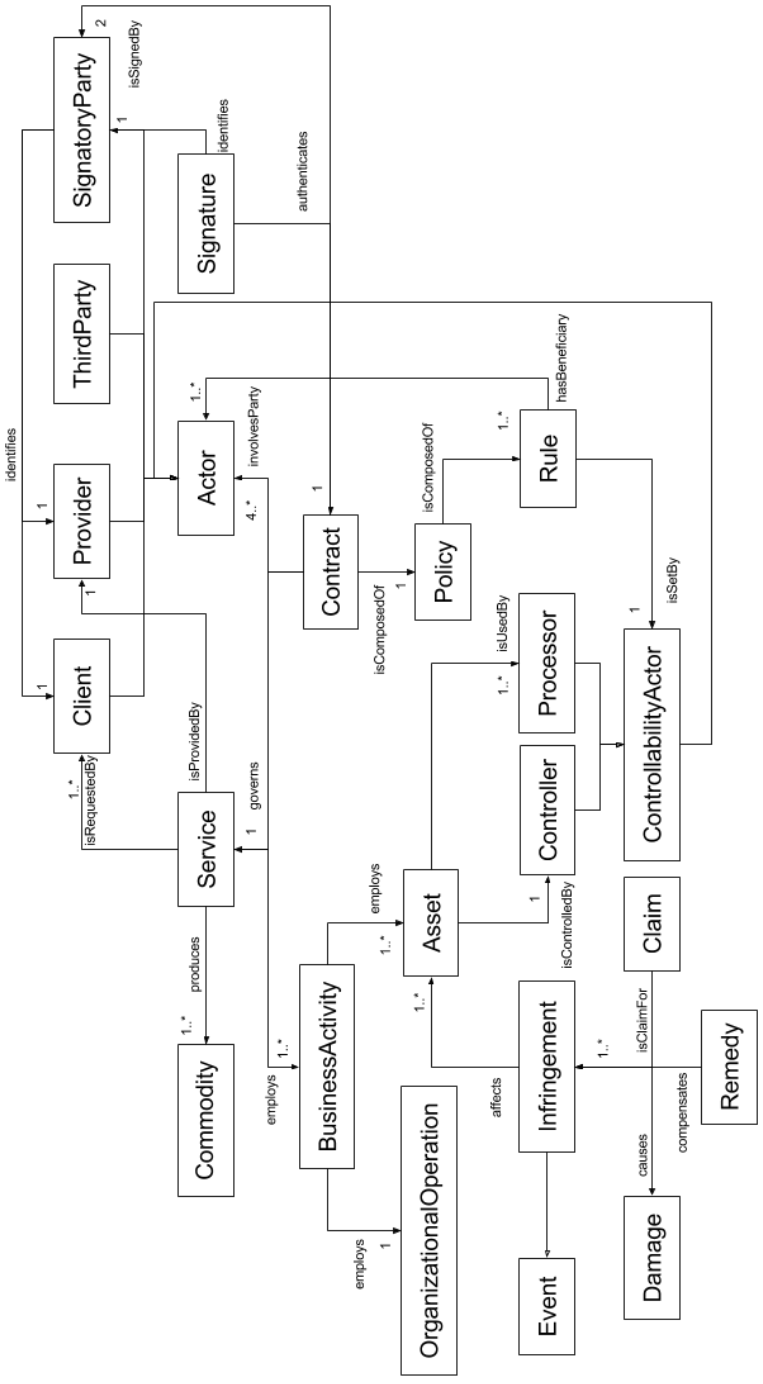


Figure 3.12: Semantic Contract Model

3.5 Machine-readable Semantic Contract

As it was seen from the previous section, the DL-based representation allows to create models supported in the logic formalism to describe knowledge belonging to a specific domain. Its advantages are the ability to make inferences as well as to serve as foundation for specific machine-readable representations. Currently, two main semantic languages are used to translate the semantic representation into a syntactic one, namely Resource Description Framework (RDF) and OWL.

RDF was the first language used with the purpose of creating machine-readable annotations that represent human knowledge about the content and meaning of data. Knowledge representation in RDF is based on the definition of triples composed of three elements: a subject(S), a predicate(P) and an object(O), written as P(S,O). Subjects and objects are concepts belonging to the domain of knowledge to be represented, and the predicate expresses relations that associate subjects and objects among them. Particularly, the subject represents the concept wanted to define and the predicate and object form the definition itself, i.e. what one wants to say about the subject. In RDF, the semantics is achieved by establishing relations among concepts. To illustrate this, let us take as example the definition: a contract is a legal document. This definition will be represented in RDF as a triple taking the form *is(Contract, LegalDocument)*, where *is* plays the role of predicate, *Contract* is the subject and *LegalDocument* is the object. Such a definition can be further refined by relating *Contract* or *LegalDocument* with other concepts. Due to this chaining, triples could be represented as labelled graphs where the nodes are the concepts of the domain of knowledge (subjects and objects) and the edges are the predicates (Figure 3.13).

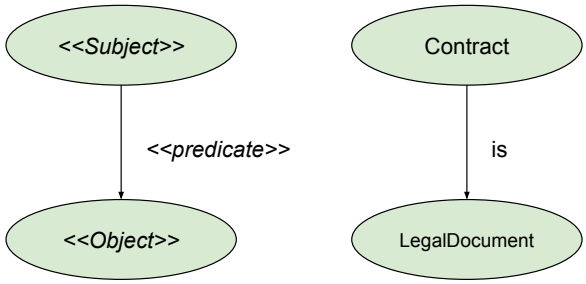


Figure 3.13: RDF Graph Example

Even if this representation is powerful, it has a constrained expressiveness in particular to hierarchies and restrictions on the predicates. To overcome some of its limitations, RDF Schema (RDF(S)) was included to the RDF representation, adding new properties to its expressive power such as the definition of *subClassOf* (to define hierarchies of concepts), *subPropertyOf* (to define predicate hierarchies) and property restrictions (to define the domain and range of predicates).

On the other hand, OWL represents complex knowledge by defining axioms and assertions, unlike RDF, where everything is a triple. To do so, three kinds of entities are defined:

Concepts		
ServiceContract	ContractTerms	Signature
Actor	Service	ServiceAttribute
Provider	Client	SignatoryPart
Commodity	Event	Metadata
RelationalAttribute	QualityAttribute	Asset
OrganizationalOperation	ContractRule	Controller
Remedy	Damage	LegislationTerm
CommodityAttribute	Claim	GuaranteeTerm
PaymentTerm	LiabilityTerm	ClaimTerm
ActorAttribute	ThirdParty	RemedyTerm
Attribute	Processor	BusinessActivity
ExclusionTerm	Infringement	PrimaryAsset
SecondaryAsset	ContractRule	ControllabilityActor
Data	Application	ComputingEquipment
Roles		
governs	involvesParty	requests
provides	produces	employs
hasAttribute	hasClassValue	hasLiteralValue
isControlledBy	isUsedBy	uses
controls	hasBeneficiary	isSetBy
infringes	affects	isClaimFor
compensates	hasSignature	agrees
authenticates	identifies	hasValue
isAffectedBy	isComposedOf	causes
isIdentifiedBy	signs	isCausedBy
isRequestedBy	isProvidedBy	isEmployedFor

Table 3.2: Vocabulary of the Service Contracts

- *Classes*: representing concrete concepts of a domain of knowledge. In OWL, mathematical descriptions are created to express memberships of individuals to a particular class.
- *Individuals*: they are specific elements belonging of a class.
- *Properties*: OWL allows the definition of two kinds of properties, namely: object properties, which relate two classes; and datatype properties, which assign datatyped-values to relations.

Moreover, OWL adds more expressive operators than RDF to the knowledge representation by defining logic constraints and quantifiers. Thus, it allows the definition of several object properties such as functional, inverse functional, transitive, symmetric, antisymmetric, reflexive and irreflexive; as well as quantified restrictions (universal and existential), cardinality restrictions, and logic operators (union, intersection). OWL[11] is categorised into three sub-languages OWL Lite, OWL DL and OWL Full, differentiated by its expressiveness power. Although OWL Full is the most expressive one, it is however no decidable.

In order to illustrate the OWL representation, let us take as example the following sentences: "A Contract is only signed between a client and a provider. The contract myContractExample is signed between the CompanyA (client) and CompanyB (provider) in May 12th 2015 at 10:40 UTC". In this example, CompanyA, CompanyB and myContractExample are individuals of the classes ServiceClient, ServiceProvider and Contract, respectively. The property isSignedBy links the contract with its signatory parties (object property), and the date represents an attribute of the contract (data property). Finally, a restriction is represented indicating that a contract is always signed between a client and a provider. Figure 3.14 shows a graphical representation of the above example.

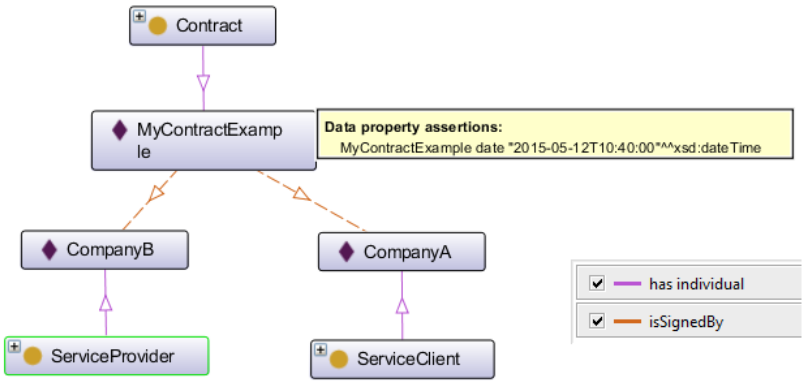


Figure 3.14: OWL Example

OWL 2 is the most recent extension of OWL which basically contains all the expressiveness of the first OWL version but improves computation, adds new properties (user-defined datatypes and property chains) and offers a new syntactic representation. Similar to OWL, it offers two major dialects to support the

[11] OWL has been also referred to as OWL 1.

semantics, namely OWL 2 DL and OWL 2 Full. Additionally, it specifies three sub-languages [66], referred to as profiles, namely: OWL 2 EL, OWL 2 QL and OWL 2 RL; which impose syntactic restrictions to improve the computational power.

Regarding the concrete representation, Figure 3.15[12] clearly illustrates the relation between the syntax and semantics for OWL 2. The RDF based semantics takes the ontology structure and map it into a RDF-graph; then the meaning is assigned to the graph. On the other hand, the direct semantics does not need to pass by that intermediate step by directly assigning the meaning to each structure of the ontology, which requires to impose some syntactic restrictions to guarantee that any structure could be directly translated into the DL theoretical model[13]. The first one is referred to as OWL 2 Full, while the second one to as OWL 2 DL, being the latter decidable. With respect to the syntax, although OWL 2 supports several concrete representations such as Turtle, Functional, Manchester, RDF/XML and OWL/XML, according to the W3C group, the RDF/XML is the only required representation to be supported by any OWL 2 tool.

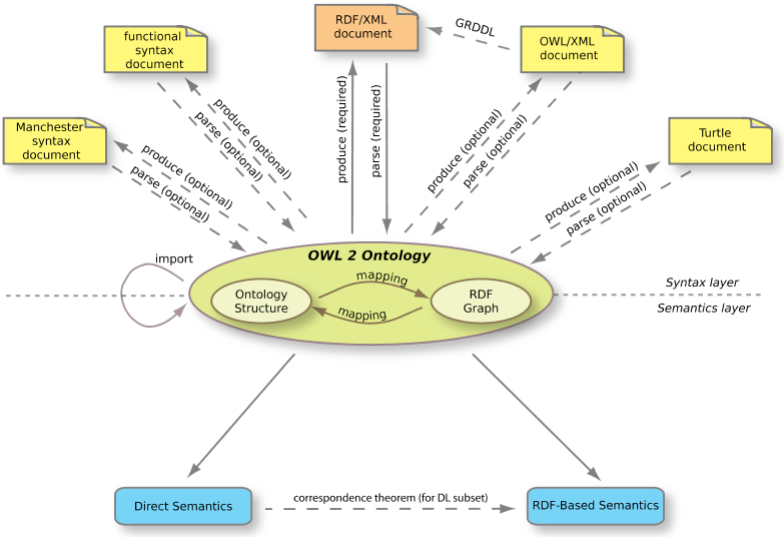


Figure 3.15: Syntax / Semantics of OWL 2

Given the needs of expressiveness, the OWL 2 language is used which supports semantics based on $\mathcal{SROIQ}(\mathcal{D})$. More concretely, the OWL/XML syntax was chose, which is a XML serialization whose aim is the interoperability with other XML representations, in particular, those oriented to the web such as WSDL, XPath, XSLT and schema-aware editors [14]. Due to the fact that service contracts are aimed to be attached to the service description, a XML representation addresses the interoperability required in the service oriented architectures. Note that the interoperation between other XML-related technologies and OWL

[12] Taken from `https://www.w3.org/TR/owl2-overview/`
[13] The complete list of restrictions is presented in `https://www.w3.org/TR/owl2-syntax/`.
[14] `https://www.w3.org/TR/owl2-new-features/`

Constructors		
IrreflexiveObjectProperty	ObjectComplementOf	ObjectIntersectionOf
DisjointObjectProperties	ObjectSomeValuesFrom	DisjointClasses
ObjectAllValuesFrom	ObjectMinCardinality	DisjointUnion
AsymmetricObjectProperty	ObjectExactCardinality	SubClassOf
FunctionalObjectProperty	InverseObjectProperties	ObjectUnionOf
ObjectPropertyDomain	ObjectPropertyRange	EquivalentClasses
SubObjectPropertyOf	DataPropertyDomain	ObjectOneOf
TransitiveObjectProperty		

Table 3.3: Contract Ontology Constructors

could be leveraged by the use of GRDDL transformations; also, some toolkits exist for the translation of OWL/XML into RDF/XML.

Considering that the way in which the syntax is mapped into the semantics is part of the reasoner implementation, the proposed semantic contract representation was validated against the Hermit 1.3.8 reasoner to guarantee both the decidability in inference tasks and that the represented knowledge is not contradictory. HermiT has been developed to tackle direct semantics, therefore it is fully compliant with the OWL 2 DL. [111] presents the comparison of several OWL 2 engines regarding their performance against some reasoning test cases.

The machine-readable implementation of the semantic contract model was done by mapping the concepts and roles listed in Table 3.2 into classes and properties, respectively in OWL 2. The result of such mapping is an ontology described in the $\mathcal{SROIQ}(\mathcal{D})$ formalism, composed of 325 axioms (Table 3.3 presents the complete set of OWL 2 constructors used in my ontological model) and 59 classes. Figure 3.16 summarizes the result metrics for the contract ontology. The complete taxonomy of my model is presented in Figure 3.17. The complete XML representation of the semantic contract can be downloaded from [106].

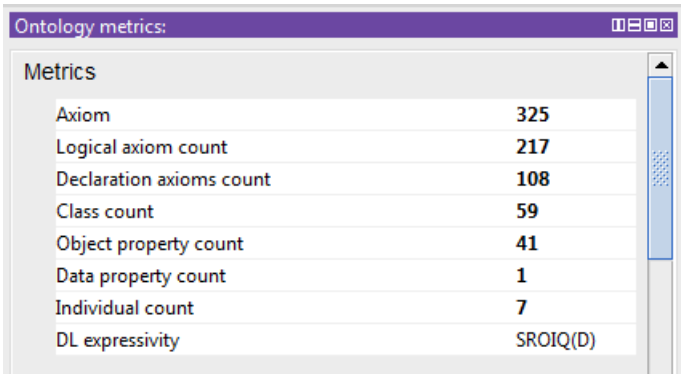


Figure 3.16: Contract Ontology Metrics

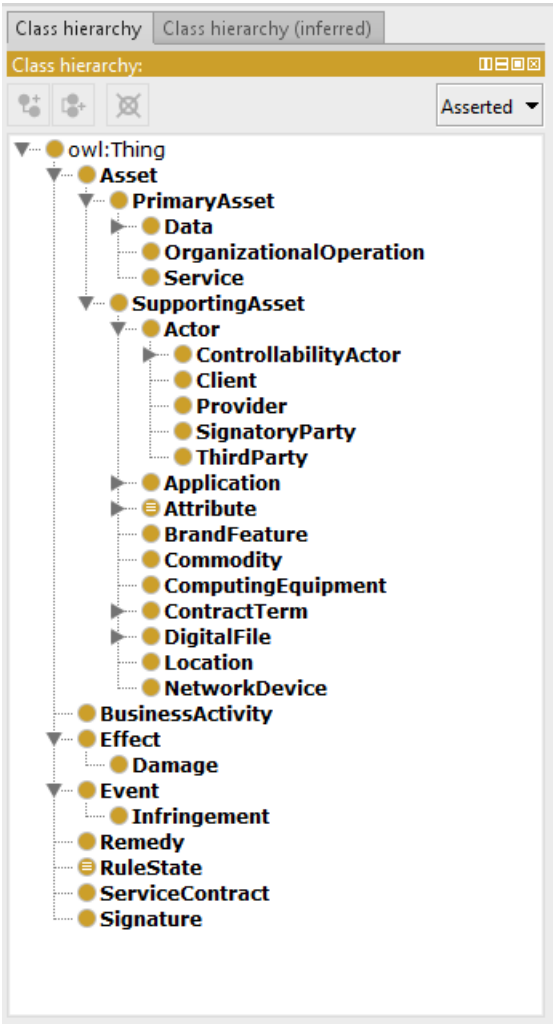


Figure 3.17: Taxonomy of the Contract Ontology

In order to illustrate the validation of the proposed semantic contract in real cases, let reconsider the excerpt of contract presented in Figure 3.3. Due to the generality of the contract model, its instantiation in a machine-readable form is done by populating classes of the semantic model with the creation of individuals. Thus, the following assertions are made:

- *OilGas_Contract* is-a *Contract*
- *OilGasCompany* is-a *Client*
- *PPMS* is-a *Provider*
- *ClientAddress* is-a *ClientAttribute*
- *ProviderAddress* is-a *ProviderAttribute*

- *EffectiveDate* is-a *ContractAttribute*
- *PublicationDate* is-a *ContractAttribute*
- *Oil_Projet_Management* is-a *Service*
- *OilGasCompany* hasAttribute *ClientAddress*
- *OilGasCompany* requests *Oil_Projet_Management*
- *PPMS* hasAttribute *ProviderAddress*
- *PPMS* provides *Oil_Projet_Management*
- *ClientAddress* hasLiteralValue $1234 NE 12 Ave, Miami, FL, 33135, USA$
- *ClientAddress* hasLiteralValue $12 Rue Blue, 4324, Aquitaine, France$
- *EffectiveDate* hasLiteralValue $2015-08-12$
- *PublicationDate* hasLiteralValue $2015-08-01$
- *OilGas_Contract* hasAttribute $EffectiveDate$
- *OilGas_Contract* hasAttribute *PublicationDate*

With the above assertions, the machine-readable representation of that excerpt of service contract is presented in Listing 3.1.

Listing 3.1: Machine-readable representation for a contract excerpt

```
<owl:NamedIndividual rdf:about="#OilGasContract">
        <rdf:type rdf:resource="#ServiceContract"/>
    <contract:governs rdf:resource="#Oil_Projects_Management"/>
    <contract:hasAttribute rdf:resource="#EffectiveDate"/>
    <contract:hasAttribute rdf:resource="#PublicationDate"/>
    <contract:involvesParty rdf:resource="#PPMS"/>
    <contract:involvesParty rdf:resource="#OilGasCompany"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="#ClientAddress">
        <rdf:type rdf:resource="&contract;ActorAttribute"/>
        <contract:hasLiteralValue rdf:datatype="&xsd;string">
       1234 NE 12 AVE, Miami, FL 33135, USA
    </contract:hasLiteralValue>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="#EffectiveDate">
    <rdf:type rdf:resource="&contract;ContractAttribute"/>
        <contract:hasLiteralValue rdf:datatype="&xsd;dateTime">
        2015-08-12T08:10:00Z
        </contract:hasLiteralValue>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="#Oil_Projects_Management">
        <rdf:type rdf:resource="&contract;Service"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="#PPMS">
        <rdf:type rdf:resource="&contract;Provider"/>
```

```
    <contract:hasAttribute rdf:resource="#ProviderAddress"/>
    <contract:provides rdf:resource="#Oil_Projects_Management"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="#ProviderAddress">
        <rdf:type rdf:resource="&contract;ActorAttribute"/>
    <contract:hasLiteralValue rdf:datatype="&xsd;string">
        France at 12 Rue Blue, 4324, Aquitaine
    </contract:hasLiteralValue>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="#PublicationDate">
    <rdf:type rdf:resource="&contract;ContractAttribute"/>
    <contract:hasLiteralValue rdf:datatype="&xsd;dateTime">
            2015-08-01T00:00:00Z
    </contract:hasLiteralValue>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="#OilGasCompany">
    <rdf:type rdf:resource="&contract;Client"/>
    <contract:hasAttribute rdf:resource="#ClientAddress"/>
    <contract:requests rdf:resource="#Oil_Projects_Management"/>
</owl:NamedIndividual>
```

The graphical representation of this contract header is presented in Figure 3.18

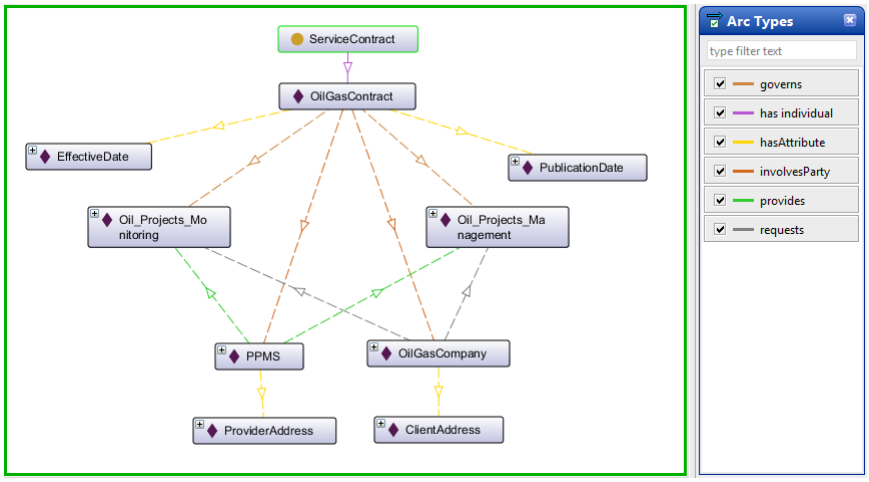


Figure 3.18: Graphical Representation of a Contract Header Example

3.6 Discussion

One important aspect for an organization is the assessment of the service. Traditionally, the Quality of the Service has been associated to infrastructure and performance properties. Such a definition is appropriated when the core of the business lies in information systems, as in the case of the cloud. If it is not the case, some other high-level and business criteria need to be considered, which

should ideally reduce the gap between the actual provision of the service and the client's expectation in order to define a service as being of quality.

Psychological contracts [47] have been a well studied field in marketing and management. Basically, they are based on the belief that contractual parties have entered some mutual obligations, which, under the principle of reciprocity, result in a relation where a party makes something and the other party expects something else in return. Although those contracts do not have any legal value, they affect the organization in aspects such as reputation, good name and business image because some discrepancies lie among expectations, obligations and the reality. Hence, to prevent risks due to misunderstandings, it becomes fundamental to clearly define the terms governing the collaborative relation between clients and providers, which will also serve as a baseline for the establishment of the criteria used for the evaluation of the service and its involved actors. However, capturing and explicitly representing contractual rules is not a trivial issue since contracts, as any other social representation, can have different interpretations. Rousseau's works [98] [97] has proved that if a contractual party has the perception that expectations are not fulfilled, the person's behavior is modified by lowering the perception about one's own obligations. Therefore, I affirm that the provision of the service is not only about the commodity itself but also about the way it is provided, i.e. the actor behavior. All together makes part of the assessment of the service quality and the definition of attributes such as the actor's reputation. Hannah [47] expresses this by stating that in the service provision is "not only deliver what it is promised, but also more importantly strives to deliver what the consumer thinks it promise".

Such a notion coming from the marketing and management domain is also pertinent in the technological perspective of the service provision since it illustrates what is relevant to model in the contract relation between clients and providers. In this approach, I propose that those expectations be reflected in a contractual policy, whose machine-readable form allows to automate some processing tasks aiming at decision-making within organizations.

In this chapter, the knowledge in the domain of service contracts was represented as meta-information which will be used to assign semantics to the definition of policies regarding the use of some organizational resources. In general, the use of ontologies to represent the knowledge involved in contractual relations is not new, neither is the representation of knowledge in the service provision in SOA. The SOA-O 2.0, proposed in 2014 by The Open Group [45] as a standard ontology for SOA, aims to fill the gap between the business and IT vision of a service-oriented architecture. In SOA-O, service contracts are "agreements needed in order to define how to use a service. [...] A service contract is binding on all participants in the interaction, including the service itself and the element that provides it for the particular interaction in question" [45]. Although this definition agrees with the proposed vision of service contracts, the functional elements (components) of the SOA domain constitute the core of that standard; then no ontology for service contracts is proposed. Other more general approaches of contract ontologies are mentioned in [93] [29] [116] and [60]. In [116] a contract ontology is presented mainly based on a taxonomy of contract concepts. This work explicitly states problems that arise in a contractual relation and classifies them in pre-contract problems, contract-phase problems and post-contract problems. Since that approach is not focused on service-oriented architectures, the loss of control in assets is not included in the

categorization of contractual problems. Moreover, no analysis is proposed regarding the business perspective of the service provision as part of the contract terms. The authors of this work highlight the importance of creating ontologies based on the expressiveness capability of OWL instead of the expressiveness of the natural language. Another approach of a contract ontology is presented in [29], where the authors state that an ontological model should map as accurately as possible the reality of the world, otherwise the model could represent "concepts conceived as human creations". Based on a philosophical view, this work proposes a perdurantist ontology of contracts, according to which an object is defined by attributes partially present during the object's existence. In this approach, a contractual party is completely defined by commitments, states and execution events. Despite the fact that some analysis are made addressing the duality of events, the reciprocity of commitments and the description of the economic resources (i.e. the exchanged resources) are not explicitly modeled as a class or type of element. Moreover, neither formal definition of the commitments is presented, nor any concrete representation.

In general, to the best of my knowledge, contract ontologies are mainly used to create taxonomies and an agreement in the vocabulary, instead of annotating data with its semantics meaning. As expected from the literature review presented in Section 3.2, those ontologies do not cover a controllability vocabulary useful for the definition of contractual policies.

On the other hand, security ontologies [93] [39] count on their vocabulary key terms such as *asset* and *control*, whose semantic representation is guided by the ISO 27001, COBIT 5, and the German IT Grundschutz Manual. Consequently, although no description of those concepts in terms of contractual obligations is proposed, the guided references they use make the proposed definitions of organizational assets match the top level; however a different but not contradictory representation is proposed in the subsumption axioms since while here assets are decomposed in two categories, namely: primary assets and supporting assets, Ramanauskaite's work proposes tangible and intangible resources are direct subclasses of the class asset. Regarding the definition of *controls*, they are instantiated as individuals (representing business measures which mitigate risks) instead of a set of business rules imposing restrictions to some people. Finally, unlike this work, threats and the vulnerability of resources are explicitly defined. In that work, individuals are assigned to classes allowing to represent if a particular threat is deliberate or accidental. In this approach, on the contrary, I seek to carry out inferences aiming to determine what happened during the service provision instead of creating assertions. However, despite the efforts in the knowledge representation, it is not possible to infer whether a fault is accidental or deliberate. Briefly, both Ramanauskaite's and Fenz's work present a comprehensive vocabulary when its represented knowledge remains inside the organization, since the description of some concepts can compromise the organization if the ontology is used for annotation purposes in inter-organizational environments.

In short, no current ontology is enough complete to represent the knowledge of controllability of assets in the frame of service contracts, requiring more precise relations which capture the behavior and the dynamism of a service provision as well as contractual relations. However, it should not be forgotten that one of the design criteria of an ontology is extendability. In particular, from the above works described, it is noted that the presented model could

Proposed Semantic Model	SOA Ontology	Security Ontology
Policy	Policy	
Actor	HumanActor	
ServiceContract	ServiceContract	
Service	Service	
Effect	Effect	
Asset		Asset

Table 3.4: Concept Equivalence at the Top Level with Existing Ontologies

be integrated to the SOA-O ontology, improving its expressiveness by adding new concepts which represent a clear understanding of the components of a service contract. Similarly, the work presented in [93], which is a mapping of Fenz's work [39] with other security standards can be refined with concepts of the contract vocabulary and new relations for the definition of controllability as a new security property as proposed in [68]. Such a representation will allow to capture the knowledge existing within organizations about the expected use of assets to control and limit harmful effects due to possible misuses of some organizational resources. In both cases, the ontology integration will help enrich the expressiveness of existing approaches and carry out more meaningful queries to the knowledge base.

In general terms, integrating two ontologies implies matching two semantic related vocabularies by finding correspondences between entities while keeping the consistency of the assertions and axioms of each individual representation. This process is a challenging task since the mapping should consider semantic and syntax representations; however such an integration aims to gain in expressiveness and precision, which justifies the fact that several approaches based on the performance, syntax, semantic, language and methodology have been proposed. Complete surveys of ontology matching and a comparison of current available supporting tools are presented in [21][62]. In general, the mapping process seeks to align ontology entities, i.e. classes, objects and properties in Ontology_1 with classes, objects and properties in Ontology_2 respectively. In [95] [113] a declarative language called CAR is suggested which is able to map an object from one entity category on another one, for example, attributes on classes, relationships on classes or relationships on attributes. Similarly, in [17] the Distributed Description Logics (DDL) formalism is proposed to create bridges rules which represent inter-ontology axioms and allow to keep the independence of each integrated ontology. Although the integration of the proposed semantic model with current ontologies is out of the scope of this dissertation, and is proposed as a future work, some concept equivalences can be manually established. In Table 3.4 equivalences at the ontology upper level are presented, where the first column contains the concepts of the presented model, the second column the SOA-O ontology [45] and the third column the security ontology presented in [93] (which is an extension of Fenz's work [39]). Moreover, some subsumption axioms can be established, for instance between the *Organization* security concept and the *Client* and *Provider* service provision terms.

Besides the ontology integration, a multi-level approach can also be con-

sidered which supports the needs of knowledge representation and inferences of both intra- and inter-organizational environments. As previously stated, [93] presents a comprehensive security and risk management vocabulary, but its level of detail has as side-effect that it is not suitable in inter-organizational environments when the ontology is used for annotation purposes, although its representation really meaningful for the organization. Therefore, it may be useful to split the knowledge representation into two layers: a shared inter-organizational ontology delivered when the service is requested, which explicitly establishes the semantics of the collaborative policies; and in the top level a more expressive representation which describes threat and vulnerabilities of the assets described in the inter-organizational policy (Figure 3.19).

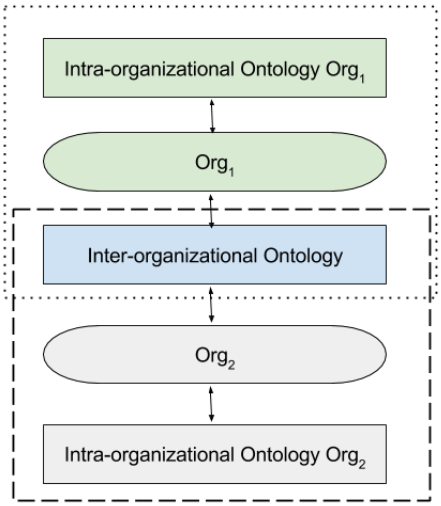


Figure 3.19: A Two-layer Semantic

Another important aspect within the knowledge representation besides extendability is querying. Indeed, some useful knowledge can also be discovered by querying the knowledge-base about the individuals belonging to a particular class (instances of a concept) or about the relation between two classes. For instance, listing all the assets involved in the service provision or retrieving the purpose of some contractual rule. Those tasks are mainly supported by the SPARQL language, which is a W3C Recommendation. A SPARQL query is similar to any relational database query such as SQL and provides operations such as joint, sort, aggregate. The query presented in Listing 3.2 retrieve all the contractual parties of the service contract extract presented in Figure 3.3.

Listing 3.2: SPARQL to List the Contractual Parties

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX
 inst:<http://www.controllability.com/exampleOilGas.owl#>
PREFIX
 contbase:<http://www.controllability.com/contract.owl#>
```

```
SELECT ?party
        WHERE {inst:OilGasContract contbase:involvesParty ?party}
```

Finally, I strive to mention a last element concerning the knowledge inference: the Open-World Assumption (OWA). This is an important assumption of the semantic web languages such as OWL. Under the OWA, any knowledge that is not explicitly asserted or inferred is considered as *unknown* regardless if it is true or false in the universe of the domain of knowledge. This is opposed to the Close-World Assumption (CWA) which considers that all the knowledge is available, so if some knowledge is not known (asserted or inferred), then is false. To illustrate the OWA, let us suppose that the concept *Attribute* is defined with the following primitive axioms

$$Attribute \sqsupseteq QualityAttribute$$
$$Attribute \sqsupseteq RelationalAttribute$$

and that a quality attribute is defined as

$$\{Attribute \wedge \neg RelationalAttribute\}$$

Under the OWA, if an individual a is asserted as an *attribute* but not a member of the class *RelationalAttribute*, the reasoner's answer to the question *Is "a" member of the class QualityAttribute?* will be *unknown* since that knowledge has not been explicitly asserted or inferred. Such an answer is justified by the fact that it is possible that the class *Attribute* be composed of other subclasses apart from *QualityAttributes* as illustrated in Figure 3.20. This has serious implications in the reasoning process since in some cases (as this one for example) we are interested in representing that any attribute which is not relational, should be automatically classified as an attribute of quality. The good news is that OWL has some operators which delimit the reasoning process based on the Open-World Assumption. First, as illustrated in Figure 3.5, a covering axiom should be asserted to represent the fact that the class *Attribute* is only composed of the subclasses *QualityAttribute* and *RelationalAttribute*. Moreover a closure axiom should restrict the property *hasClassValue* to be filled only with individuals belonging to the contract definition and not with data-properties; similarly, the property *hasLiteralValue* needs to be filled by single data-typed elements only. It is expressed in the OWL syntax as *hasClassValue* **only** *owl:Thing* and *hasLiteralValue* **only** *xsd:anySimpleType*.

Similarly, it has been established in the literature that the OWA has its justification from the need of having monotonic inference. In this model, as stated in Section 3.4, the lack of OWL operators and DL mechanisms to retract asserted knowledge was approached in this model by representing mutability attributes as individuals instead of data or object properties.

3.7 Conclusion

The ontology-based representation has shown its advantage to share and formalize knowledge in a machine-readable form as well as to draw conclusions from the asserted knowledge. In this chapter, it has been shown that current approaches to represent machine-readable service guarantees have a limited expressiveness to describe coarse-grained business rules, notably, rules about the

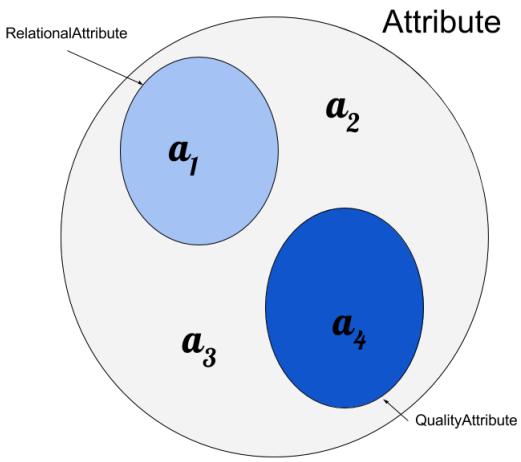


Figure 3.20: Open World Assumption Example

use of assets. Therefore, I propose to improve the expressiveness of service agreements through the creation of service contracts. As a first contribution of this dissertation, a formal semantic contract model based on the DL formalism is proposed.

The semantic contract specifies a vocabulary composed of 59 concepts which model the relation between clients and providers of a given service. In such a model, concepts are defined generally considering the provision of any service from a business perspective instead of a technical one. Similarly, besides the quality attributes, relational attributes are proposed which overcome the limitation of existing SLA approaches which only represent attributes as a parameter-value relation. The proposed model represents a vocabulary of the elements involved in the definition of policies aiming at the control of organizational assets, in the particular case of a service provision. An important implication of the definition of such a vocabulary is that it agrees with the semantics of the rules governing the collaborative relation between contractual parties by giving a common understanding of their meaning, since it has been noted that such a rules are usually written by using some business terminology specific to each organization. Similarly, the relevance of the proposed representation has been highlighted to improve existing approaches. On the one hand, the mapping process of the semantic contract model on security ontologies would contribute to add a new dimension to the definition of security by including controllability as a new attribute, according to which the sharing of assets, although unavoidable in the service provision, can represent organizational risks due to the harmful effects of possible misuses of some organizational resources. On the other hand, from the point of view of service-oriented architectures, the proposed model gives a description of the role of the functional SOA elements in the context of contractual relations. In the following chapter, I will describe the way in which the proposed semantic representation will be used as meta-information to annotate the policy governing the service provision.

Chapter 4

A Model for the Controllability Terms based on Rules

The definition of a contractual policy, governing the interaction between clients and providers, allows to clarify the commitments and expectations of each party with regard to the service provision. In the previous Chapter, the semantic of the elements involved in the service provision, including the definition of a controllability vocabulary was formalized by the DL logic. In this Chapter, I will propose a model to formalize controllability requirements as part of non-functional business guarantees governing a service provision. In particular, due to the fact that those requirements are usually part of the internal policies of an organization, and are defined using a business vocabulary, I will use the semantic model defined in Chapter 3 to avoid misunderstanding in their meaning. The validation process of the proposed model is presented through the implementation of a policy in XML taken as basis an excerpt of two real contracts.

Objectives

- To propose a model for the controllability requirements which be able to represent the expected behavior of an external partner regarding the use of assets.

4.1 Introduction

In the business to business approach, the client's decision to incorporate into a business process a service provided by an external organization is based on the principles of delegation and trust. Indeed, the client delegates an activity or a process to an external partner, who offers the required business functionality

as a service. Also, during their interaction, some assets[1] are exchanged and shared, where each party can only trust that one's counterpart will act in good faith with regard to their usage.

In order to explicitly establish the expected behavior of each party, contractual terms are set, clarifying the criteria that govern the relation between clients and providers. Those terms become the actual binding between the contractual parties. By signing the contract, bound parties agree on the contractual terms and they commit to provide the service by respecting them. Thus, in case of auditing or claim, the contract terms are used as a basis to evaluate both the service provision and the behavior of the contractual parties, where the consequences of any deviation of an agreed contractual term may range from economic compensation to the avoidance of the contract, depending on the type of the infringed term.

Contract terms, regulating a service provided by an external party should be able to represent not only functional aspects about the way in which the service is offered, but also to provide enough information that makes the expected use of the delegated and shared assets explicit. Thence, the terms of a service contract may specify guarantees about some attributes of the commodity, attributes of the service or its involves parties, as well as the way in which the contract or the service can be modified, or the path to follow in case of non-compliance with a contract term. In this approach, it is assumed that those objects on which the contractual parties aim to keep control of are part of organizational assets. Similarly, the behavior to be restricted (controlled) defines the expected use of such an asset. From the organizational point of view, the appropriate definition of the contractual terms plays an important role from the risk management perspective, in view of the fact that:

- Contractual terms represent the commitments of each party. Therefore, they contribute to avoid mismatched expectations about the service provision or the behavior of each party.
- The terms defined in the service contract can be used to resolve disputes in case of litigation.
- The evaluation of the accomplishment of the contractual terms can be used to determine the client satisfaction as well as the quality of the provided service.
- Contract terms should represent the information needed (i.e. the description of the expected external partner's behavior) for the organization to protect itself from potential damages due to the exploitation of vulnerabilities or misuses of assets.

As concluded in Section 3.2, even if the definition of contract terms has showed to be highly valuable within organizations, its current representation is restricted to plain-text documents (digital or paper-based), while only a subset of them, such as security assertions and monetary terms are represented in machine-readable agreements. Previously, I have presented in Chapter 3 the

[1] As it has been defined in the previous Chapter, such assets could be data, information, equipment, software, activities, and without going very far, the delegated business process itself.

main concepts involved in the formal definition of contracts and controllability; I have specified their semantics by creating relations among those concepts using a $\mathcal{SROIQ}(\mathcal{D})$ formalism. Thus, for example, when one talks about a provider, one knows that the person is the actor who provides the service, and who can control and use some assets. In this chapter, I will use those definitions to add a formal semantics to the components involved in the contract terms. In this chapter, I will present a model for the definition of the terms of the contract representing high-level business wishes and requirements about the service provision, particularly, those imposing restrictions on the use of assets. This chapter is organized as follows. In Section 4.2, I will present the state of the art about the representation of policies in a machine-readable form. Subsequently, in Section 4.3 I will introduce the knowledge representation based on rules and its underlying formalization by using a subset of the FOL. In Section 4.4, I will describe my proposed model for the definition of the controllability requirements as part of the contractual terms governing the service provision. Later, its implementation in a machine-readable form is illustrated with an example in Section 4.5. Finally, in Sections 4.7 and 4.8, a discussion of the proposition, the perspectives and conclusions will be presented.

4.2 State of the Art

In this Section, current approaches aiming at the modeling of organizational policies are presented. The reviewed works have been grouped into two categories: works modeling policies through the creation of rules based on a subset of the FOL, and policies which integrate semantics in the definition of rules.

Syntax-based policies

Looking for the modeling of policies mediated by the information systems is equivalent to investigating in security policies. Indeed, security, and more concretely access control is the most accepted approach to tackle risks within organizations. Therefore, in order to protect the organization against damages, the use of the information system is controlled, where such an use refers to its access. This kind of control prevents that unauthorized people access systems by targeting the protection of one particular kind of asset, i.e. the data. Following, an overview of the main security policy models is described. Then, a description of the recent models proposed to tackle the creation of policies in inter-organizational environments will be presented.

In the most general classification, access control policies are divided into four categories, namely: mandatory control, discretionary control, role-based control and rule-based control. First, the mandatory access control model [101] is based on subjects, subjects' label, objects and objects' label. In general, for each request, the system grants the access by evaluating the label individually assigned to the subject and the label of the object (representing the sensitivity of the object to be accessed). Secondly, the discretionary access control model [31] includes an Access Control List (ACL), the object and its owner. Basically, the authorization rights are assigned by the owner of the object to a subject or a group of subjects, who can pass granted rights to other subjects who inherit his authorizations. As its name suggests, the main component of the third category

is roles. The authorization based on roles can enforce any of the two previous approaches, and it proposes a high-level layer to reflect the fact that authorizations are not directly granted to individual subjects, but to roles; the subject inherits those permissions when assigned to a role [40]. This model is suitable for large organizations for which the manual assignation and modification of rights become a hard operation. Finally, the access based on rules seems to be the most dynamic one since it aims at the creation of rules describing more complex restrictions for the access authorization. OrBAC [61] is a model based on the FOL (concretely, a Datalog program is proposed in order to tackle undecidable results), which formalizes the conditions under which some right is granted by proposing the inclusion of contextual information within the definition of access rules. The core of OrBAC is the concept of organization, which links the two layers proposed in its model. The first one corresponds to an abstract representation which is composed of the definition of roles, activities and views, while the second one defines the concrete entities for each abstract category, namely: subject, action, and object, respectively. This last layer allows the specification of the conditions (context) under which the rule should be triggered.

In the last years, the growing interest in inter-organizational collaboration and the widely adoption of distributed architectures, such as SOA, have highlighted the need to having organizational policies able to govern the interaction between independent actors[2]. The work presented in [12] contributes to tackle this need. The proposed approach is able to support both intra- and inter-organizational workflows. Arguing that the access policies should be in line with the workflow execution, the author proposes the transformation of a policy created according the OrBAC model into a Petri Net able to manage the workflow execution. This approach combines both the access control and the flow control. Although the underlying proposition for the definition of policies relies on OrBAC, it offers the formalization of a predicate to represent the changes of roles performed during the flow execution. The work presented in [15] also describes an interesting approach based on the definition of policies directly attached to data. This approach inherits the definition of policies of XACML, and it is also able to describe the purpose for the data handling as well as the obligated actions to carry out after the data collection. Despite the fact that this work adds interesting features in the definition of policies, it does not directly guarantee that the policy is enforced, so they introduce a tamper-proof engine, which is certified by a trusted third-party. FI-OrBAC [70] is another organization-based model which aims at the access control for a federated identity platform. This work, based on the FOL formalization, assumes the existence of a trust domain in which an area resources are accessed by subjects belonging to another area. In this model, the subject attributes are evaluated to assign rights. Such attributes are divided into persistent attributes and evolution attributes. Moreover, to gain in expressiveness, three predicates are proposed: frame (used for the definition of a matrix of attributes), delegate and detain. The work presented in [99] also bases its representation in the Datalog subset of the FOL. This approach models new policy elements such as Activity_in_Organization and UserAccounts. This latter is highly important for the model description since policies are not tied to subjects but to user accounts. Regarding the expressiveness of rules, the proposed approach models obligations, interdiction, permission, pre-prohibition

[2] The term actor is used here to encompass both individual subjects and organizations.

and pre-obligation, where the last two predicates requires the definition of a mutable weight to order the access policies. Therefore, repetitive violation attempts lead to changes in the weight values in a way that an obligation may become a prohibition. Finally, usage control policies (UCON), and its variants UCON_{ABC}, ConUCON and CA-UCON, [76] [4] [90] propose a well formalized model based on mutability of subject and object attributes. Unlike the aforementioned approaches, UCON aims at the continuity of the access decision; it means a granted right can be revoked based on the contextual information. In order to guarantee mutability and continuity, the execution of an access request moves through several states, namely: requesting, pre-adapting, accessing, on adapting, revoked and denied. In its simplest form, the UCON model is based on subjects, subject attributes, objects, object attributes, rights, authorizations, obligations and conditions, while some other elements have been added according to their variants such as the context entity, event, temporal constructors, pre-authorizations, pre-obligations, pre/post-conditions, ongoing authorization, ongoing obligation and ongoing conditions.

Unlike the previous works focused on access-based policies, the work presented in [88] proposes the use of Logic Programming (LP) to create a Rule--Based Service Level Agreement Language (RBSLA). The author states that service agreements "should make the contractual rules explicit in a formal, machine-readable and interchangeable way in order to transform them into executable code and exchange them between business partners and organizations". Thus, this work highlights the lack of formalism and semantics of web service policies (such as WS-Policy) since they are based on procedural and imperative logic, and proposes the implementation of rules which are expressed in a machine-readable syntax using the RuleML language. RBSLA is able to express deontic norms, defeasible rules, rule priorities, SLA domain specific elements such as metrics, escalation levels, post-conditions, and according to the author it will also be possible to integrate RDF/OWL vocabularies although no specific details are described. Regarding to the controllability problem, it is a very interesting work since, similar to ours, it highlights the importance of explicitly establishing the conditions of a service agreement, in both machine-readable and formal way. In addition, this work presents a high expressiveness to represent rules. However, the object of such a rules is focused on traditional SLA metrics, i.e. hardware, software, network and storage.

Semantic policies

Unlike the previous works in which the semantics of a concept is done by the interpretation of the meaning of a predicate or a XML tag, other approaches aim at a more complex definition of the elements of the policy by the inclusion of semantics. It allows to integrate the advantages of a semantic representation such as reusability, disambiguation of the definitions, and an easy sharing, which makes it suitable for the definition of inter-organizational policies. In [27], it is stated that policies based on data identifiers and simple predicate are not enough to grant rights to a subject. Therefore, a modification of the XACML language is proposed to tackle semantic descriptions of subjects and objects described in the policy. XACML is an OASIS standard for the definition of attribute-based security policies in XML. The authors of this work overcome the limited

expressiveness of XML to describe complex attributes by the inclusion of reified RDF statements in the `AttributeValue` tag of XACML. The works presented in [28] [26] also address the inclusion of semantic descriptions to the definition of access policies. Those works have proposed the inclusion of semantic aware contexts and have analyzed the effects of concept hierarchies and mereology for the access, notably in terms of modality conflicts.

In [23][20] the advantages of semantic descriptions are also highlighted but this time for the description of business rules. In this work, the Semantic of the Business Vocabulary & Business Rules (SBVR) is translated into an OWL + SWRL knowledge base. The authors of this work state that the DL formalism, although very expressive, is not able to describe complex knowledge involving the relation of several concepts at the same time. Therefore, they propose the use of rules based on the Horn logic which are implemented using the SWRL language. This work shows the usefulness of combining two formalisms, description logics and Horn logics, to create more complex description of concepts, specially when those descriptions are subject to conditions.

In general, from the literature review, it is concluded that more effort is needed to fill the gap between the business and the technological view of the organization's requirements. It is reflected in the imbalance between security policy works and business policy works. Indeed, securing the information system is not the only way to protect the organization against damages. Moreover, data is not the only organizational asset exchanged during a service provision. In light of the controllability problem, current security models, including usage control models, focus on access control. Their limitation is to be based on atomic and observable actions (reading, writing, download, creation and deletion). They assume, for the inter-organizational policies based on workflows, the existence of a scenario in which all the knowledge about the processes is known, which is surely true for a particular actor (he has all the available information about his internal processes) but not for a service provision as a whole. The implication of consider only concrete and single actions relies on policy enforcement. In access control policies, the Policy Enforcement Point (PEP) and the Policy Decision Point (PDP) work together to determine compliance with the policy, on the basis of events captured by the information system (attempt to access, attempt to modify, etc...) making decisions in runtime about the rights to be granted. However, business activities are neither always observable nor atomic, which requires the modification of the policy model and its inherent enforcement mechanism. As previously described, inter-organizational policies are system-centric; it means, it is assumed that external partners are authorized to use (access rights) assets that remains in the controller side of the collaborative relation. However, when the asset leaves the controller, as in the case of the data with attached policy, the current mechanisms to enforce the policy are based on trust. Despite the aforementioned limitations regarding the implementation of current access models to tackle controllability, it is clear the important contributions of those approaches in terms of formalization and modeling. Particularly, with regard to the expressiveness of the underlying formalism and the elements to model such as the inclusion of weights and temporal operators.

On the other hand, business rules based on LP and Horn logic are able to express coarse-grained activities instead of single actions; due to the fact that they also rely on a subset of the FOL, temporal operators, events, and deontic operators can be included within the definition of rules. Moreover, current

languages such as RuleML and SWRL allow an easy integration of rules with semantic descriptions. However, unlike the previous group of works, the inherent model of rules does not give a clear meaning to their components. Indeed, the model of RuleML is based on operators, variables, complex terms, individuals and expressions, while the SWRL is based on atoms; nevertheless in both cases those elements are not significant in an organizational policy.

From the literature review, I have concluded that it is not a reasonable solution to use access control models by simply extending the meaning of actions to force them to cover coarse-grained activities, since the objects to which those activities apply should consequently change, as well as the mechanism for the policy enforcement. On the other hand, although semantic rules allow for a better expressiveness, a model in the context of security (controllability) policies is missing. However, works as [27] give clues about the integration of semantics with security policies.

Table 4.1 summarizes the literature review of the policy representation, highlighting their strengths and limitations from the point of view of this controllability approach.

4.3 Knowledge Representation based on Rules

Similar to the way each domain of knowledge has created its own vocabularies and models to explain and relate different objects and concepts of the world, each organization uses its own internally agreed vocabulary, associated to the core of its business logic, to define different kinds of organizational policies, in particular, policies about the use of assets. However, when an organizational asset moves out beyond the domain of control of the organization (for example, in the framework of a collaborative relation), there is no guarantee that the assets will be used following the policy established by the controller. In my approach, I infer that a subset of the organizational policies associated to the shared assets be mapped on the contractual terms, binding external partners to their accomplishment. In fact, it is a no trivial issue that an organization specify contractual obligations that need to be accomplished by other organizations. That is why I use the ontological model presented in Chapter 3 to support the definition of contractual terms. Thus, it will allow to have a common understanding of the meaning of the concepts defined within contractual terms by associating a clear semantics to them.

Contract terms are modeled according to the following criteria:

a) The terms of the contract represent a set of clauses that parties agree to comply during the validity of the contract[3].

b) Each contractual term may impose restrictions on any asset. It means, the contractual terms could also specify some restrictions based on either relational or quality attributes (for example, attributes of the service, or attributes of the commodity).

c) Different categories of clauses can be expressed in contract terms. They range from observable ones (which can be verified in real-time via a monitoring process) to high-level business clauses reflecting more coarse and

[3] Or even after the contract ending if some survival terms are defined

Approach	Strengths	Limitations
Access Control, Usage Control, RBSLA [76][4][99] [15][90][70] [12][88][61]	• Formal models. • Expressiveness based on FOL formalism. • Significant model entities in the context of security.	• Difficulty to represent complex relations for the definition of entities. • Policy enforcement in runtime. • Atomic and observable actions. • Objects are restricted to data. • Lack of semantics.
Semantic- based [23][20][28] [26][27]	• Ability to represent business rules. • Easy integration with ontologies. • Based on LP and Horn logic formalism.	• The entities of the inherent model do not have clear meaning in the context of controllability or security policies.

Table 4.1: Approaches for the Representation of Organizational Policies

abstract behaviors which can be hardly monitored in runtime by the information system. The proposed approach mainly focuses on the second category.

d) Compliance with the contract implies to prove that contractual terms are met.

Before presenting the model, let us consider the following examples of contract terms to clarify the modeling choices.

1. The Provider will delete the Client's account after the contract termination.
2. The Provider can subcontract the Service or a part of it in case of potential delay, to comply with the timeframes of the Service provision.
3. When the Client cancels the Service without notice, he/she will not be issued a refund.
4. The Services may not be used by final users under the age of 18.
5. By using the Service, the Provider undertakes to process the Client's data on the French territory.
6. By using the given Services, the Client gives the Provider the permission to store the Client's payment information, but this permission will not be extended to third parties.

First, as it was aforementioned, I have seen that restrictions can be imposed on different kind of objects: the client's account, the payment, the service, the actors' attributes or the data. Secondly, these contractual terms reflect business wishes and requirements which are difficult to monitor in runtime, i.e, could the client be sure that his/her data was deleted on the provider's side, or that data do not leave the French territory? Was the subcontracting process justified by a potential delay in the service timeframe? Are the final users of the client all of legal age to use the service? All of these questions are difficult to answer and they are even harder from a legal point of view; it is not possible to monitor all individual actions of an external party. When possible, this solution represents a security issue since it implies to open the business logic of the organization to external actors, resulting in an impractical solution. I have mainly focused on the modeling process of these kinds of contract terms.

An analysis of the semantics of contractual terms has led us to conclude that they can be rewritten as conditionals. In general, each term of the contract can be reformulated in a rule-based representation. This reformulation holds true since contract terms are themselves conditions that need to be fulfilled by the service stakeholders. This semantic analysis justifies the choice of modeling each single contractual term as a rule, where the set of contract terms defines a contract policy.

Regarding formalism behind the construction of contractual terms, the Horn clauses are a way of representing rules to perform inferences in FOL. Formally, a Horn clause is composed of a disjunction of atomic sentences (not negated predicate) with at most a positive literal, that is:

$$\neg p_1 \vee \neg p_2 \vee \cdots \vee \neg p_n \vee u \tag{4.1}$$

which is semantically equivalent to:

$$\neg(p_1 \wedge p_2 \wedge \cdots \wedge p_n) \vee u \equiv (p_1 \wedge p_2 \wedge \cdots \wedge p_n) \rightarrow u \tag{4.2}$$

In its normal form, Horn clauses formalize the conjunction of positive literals as shown in Eq. 4.3.

$$\begin{aligned}(p_1 \wedge \cdots \wedge p_n) \rightarrow (u \wedge v) \equiv & (p_1 \wedge \cdots \wedge p_n) \rightarrow u \wedge \\ & (p_1 \wedge \cdots \wedge p_n) \rightarrow v\end{aligned} \tag{4.3}$$

Contrasting imperative representations, Horn-like rules have the advantage of being supported by a well established model theory and proof theory; they are more expressive and reactive, since their flow control is not pre-determined, depending of the knowledge base and the rule base; and also more flexible to changes and compact. Similarly, it allows to easily integrate the semantic-based representation by considering the classes and properties as the literals of the Horn clauses. Therefore, in the following, a Horn-based representation of rules is considered both in its formalization and its machine-readable form.

4.4 Contract Term Model

From the literature review, it has been concluded that current service agreement approaches have a limited expressiveness to represent business requirements. First, service guarantees are not expressed in the form of parameter-datatyped value with no clear semantics. Secondly, service level metrics only represent performance and infrastructure features. Thirdly, non-functional service requirements focus on security and monetary guarantees. Moreover, it has been also shown that security models cannot be used to specify business requirements since the instantiation of rules requires the definition of concrete actions and objects which can be observable in runtime. For instance, an OrBAC rule is specified as:

$$\begin{aligned}is_permitted(Subject, Action, Object) \leftarrow\ & empower(Org, Subject, Role), \\ & consider(Org, Action, Activity), \\ & use(Org, Object, View), \\ & permission(Org, Role, Activity, View, Ctx), \\ & hold(Org, Subject, Action, Object, Ctx)\end{aligned}$$

where the predicates `consider`, `use` and `empower` are required in the definition of rules to indicate the concrete elements that are associated to each abstract activity, view and role. However, regarding controllability rules, it should be considered that the asset is out of the domain of the controller and that the external partner is considered as a black box. Therefore, it is necessary a model which allows to describe business rules at the abstract level, that is, without specifying the atomic/concrete actions carried out by the external partner, while allowing the verification of rules.

In the proposed approach, contractual terms are represented in the form of policies governing the service provision. A contractual policy is considered as nothing else than a set of rules reflecting the context-aware expectation of the given parties about the service provision. In particular, that contextual information implies that some conditions need to be verified in the knowledge base before the rule be considered as accomplished. Note that if no verification were necessary, contract terms could be easy represented as assertions in an ontological form.

As previously described, even though some languages exist for the machine-readable representation of business rules such as Semantic Web Rule Language (SWRL), they have a general-purpose representation; which is translated by the lack of significance in regards to the formalization of a controllability policy. On the other hand, although existing security models provide such a significance, they lack semantics, and their formalization of actions and objects do not apply to controllability requirements. Consequently, a meaningful model is needed to allow the identification of the elements required in a controllability rule, while being sufficiently expressive to represent business requirements. Following, I will explain each of the components of the proposed formalization. Moreover, taking in mind that the model falls in the definition of organizational policies, I will share some of the terminology of existing security-oriented policies. Therefore, I will strive to clearly identify the differences between this approach and the traditional use of some terms in access control policies.

4.4.1 Subject

The subject of a rule is an active entity which is liable of performing some activity. As previously stated, some conditions are usually defined in the contract terms to specify under which situation a term is valid. Particularly, when those conditions impose restrictions on the subject, they are modeled by the `SubjectCondition` element. Therefore, in my model, the subject of the rule is defined by two components: the concrete subject to which the rule applies to and the conditions of the subject.

$$\text{Subject} := \text{ConcreteSubject} \times \{\text{SubjectCondition}\} \tag{4.4}$$

In order to give a clear meaning to those components, I will use the semantic model to formalize the `ConcreteSubject` as an individual actor, and the `SubjectCondition` as a subset of `ActorAttributes`.

$$\text{ConcreteSubject} \in \text{Actor} \tag{4.5}$$

$$\text{SubjectCondition} \subseteq \text{ActorAttribute} \tag{4.6}$$

Note that while in the semantic model the term `Actor` is used to define any party involved in the contractual relation, in the controllability model, the term `Subject` refers to both a particular actor and some of its expected attributes. The modeling of the subject in the policy allows to determine the state of the actor at the time when some commitments are assigned (rights are granted).

Usually the definition of a subject in security policies includes both people and systems. The former defined as a requester, and the latter as any information system automatically processing some data or digital content. On the

contrary, in this approach only actors (organizations) were considered. This intentional choice is justified by the fact that external organizations are considered as black boxes, therefore no knowledge about their information systems is available, and no restriction can be imposed to them. As described in Section 4.2, collaborative security policies have been proposed which allow to define an information system as the subject of the rule; however, they are based on the principle that external subjects request the permission to (or are obligated to) carry out some actions on some resources. It means, a scenario where the resource remains within the organization, i.e. under the control of the actor defining the policy, while a set of rules restrict the permitted actions that the subjects of the external organization can perform. Conversely, in a policy aiming at controllability, it is assumed that the resource has left the organization, so fine-grained rules at the system level can not be defined; only coarse-grained rules imposing restrictions to the external partner about the way how the asset should be "externally" used.

4.4.2 Behavior

In the definition of controllability rules, the behavior reflects the way in which a subject interacts with an asset during the provision of a service. In other words, it represents the organizational needs in terms of control. In order to illustrate the proposed formalization of behaviors, let us consider the following excerpts of contract terms:

- The service client has the right to terminate the contract when...
- The personal information of the client will be deleted after...
- The provider will share the client's data to third parties only if...
- The client is obligated to send to the provider the evidences associated to the claim in order to...

From the above examples, it is identified that what is intended to bet controlled is defined by a set of organizational operations: *terminate*, *delete*, *share* and *sent*; and a set of assets being affected or used in the execution of those operations: the *contract*, the *client's personal information*, the *client's data*, and the *claim's evidences*. From the knowledge represented in the semantic model, it is seen that the relation between operations and assets corresponds to the definition of the concept `BusinessActivity`. Therefore, the activity together with the conditions restricting those activities are modeled by the element `Behavior`.

On the other hand, the use can also refer to changes in the attributes of the asset. Therefore, instead of referring to an abstract activity, the asset itself can be defined in the behavior by indicating that some of its properties are controlled. The modeling choice of specifying a concrete asset in the behavior instead of directly using the role `hasAttribute` is due to the fact that, as it will be explained in Section 4.4.5, some new attributes (not only changes in the existing ones) can be defined in rules. Therefore it will be only required that such an asset exist and expose some particular conditions.

Consequently, the behavior of the actor, which represents what is intended to be restricted (controlled), describes abstract activities involving a particular

asset, as well as the attributes of the asset; both meaning two forms of interacting with an organizational resource. The behavior is formalized as presented in Eq. 4.7.

$$\text{Behavior} := \text{ConcreteBehavior} \times \{\text{BehaviorCondition}\} \tag{4.7}$$

$$\{\text{ConcreteBehavior} \in \text{BusinessActivity}\} \oplus \{\text{ConcreteBehavior} \in \text{Asset}\} \tag{4.8}$$

In general, two main considerations are made for the definition of behaviors. First, this approach does not consider attributes on the organizational operations defined in the activity. It is due to the fact that in the semantic model, any additional knowledge to the definition of operations was included, which are only used as part of the modeling process of the business activity. Consequently, `BehaviorCondition` can only define restrictions on the assets.

$$\text{BehaviorCondition} \subseteq \text{AssetAttribute} \tag{4.9}$$

Secondly, in order to clarify the contribution of this model from other security models, I strive to differentiate activities from actions. An "action" is a term commonly used in access policies to name the rights granted to subjects. In those policies, actions are atomic, observable and monitorable. On the contrary, in this approach I employ the term activity which encompasses a higher level of abstraction to the definition of rights. The advantage of such a definition is twofold: it allows to define rules about the usage of assets by using a high-level business vocabulary, and it aims to maintain the autonomy of each organization in the way they internally perform activities. Consequently, the meaning of an activity in the current model is much broader than in its traditional sense of access. Indeed, I am moving from the definition of individual and observable actions, such as read and write, carried out on some digital content, to more coarse-grained activities performed on a large set of assets. Nevertheless, the risk of incorporating this kind of abstraction within the definition of policies is such that the verification of the compliance with the contractual terms becomes more complex. In section 4.4.8 I will describe the way in which this issue was approached.

4.4.3 Modality

In plain-text documents, the expressiveness of the natural language allows to specify the nature of the rule by using modals such as should, may, must or can. Those keywords are inherently interpreted by identifying a rule as an obligation, a permission, a recommendation or a prohibition. Such expressiveness is mapped on the model in the form of modalities. Modals are important since they allow to capture speech acts reflecting the "intention" of a contract term and to translate them into the model with the aim of being represented in a machine-readable form.

$$\text{hasModal} \supseteq \{\text{hasRecommendation}, \text{hasObligation}, \text{hasPermission}, \text{hasProhibition}\} \tag{4.10}$$

Where a particular rule cannot have two modals at the same time, i.e. a same rule can not represent both an obligation and a recommendation.

$$\text{hasRecommendation} \cap \text{hasObligation} \cap \text{hasPermission} \cap \text{hasProhibition} = \emptyset \tag{4.11}$$

An obligation rule is mandatory, its non-accomplishment is equivalent to the non-accomplishment of a contract term resulting in a breach of the contract. A permission (prohibition) represents the fact that an actor is authorized (is not authorized resp.) to do something during the validity of the contract. It should be clarified that while permissions mean that an actor *may* do something, an obligation means that the actor *must* do something. Finally, recommendations are seen as an intermediate level between a permission and an obligation.

The appropriate definition of the modal is highly important for the assessment of the contract accomplishment. Thus, the semantics of the modals implies that an obligation is infringed if the obliged party fails in its compliance, while, strictly speaking, a recommendation cannot be infringed, only ignored. Moreover, the organizational impacts resulted from ignoring a recommendation are not the same as those resulted of the non-accomplishment an obligation.

Since a controllability rule expresses what is permitted, prohibited, recommended or obligated to do if some conditions are met, it can be easily notice that the modal is part of the definition of the consequent of the rule. In order words, after some conditions have been verified in the antecedent, the modality in which the activity needs to be performed can be asserted in the knowledge base. Consequently, the modal reuses the specific actor and activity defined in the subject and behavior to assert the fact <`hasModal`>(`Actor`, `BusinessActivity`) in the knowledge base. For instance, `hasObligation`(`Provider`, `Nofity`), where the conditions for performing such an activity are defined in the rule, and the meaning of the activity `Notify` (the operation and the assets used) is expressed in the semantic model.

4.4.4 Context

In the previous sections, the conditions to define the controllability rules have been associated to the asset and the actor; however, other categories of knowledge represented in the semantic model can also be used to restrict the use of assets. Those restrictions are represented in the `Context`. The context refers to system or environmental conditions verified during the heuristic evaluation of the rule. System conditions refer to relations established within the semantic model, which determine the conditions of validity of a contract term. Therefore, the semantics of the contextual conditions indicates the state of the knowledge base at the moment of the commitment assignation. Similarly, environmental conditions reflect the particular situation do not related to the available knowledge about the service provision, in which a rule, representing a contract term, should be triggered such as the date, hour or location.

4.4.5 Behavioral and Assignation Rules

It has been stated that the modeled contract rules have an *if-then* structure, composed of an antecedent describing the conditions that need to be fulfilled to assert the knowledge represented in the consequent. The analysis on the real plain-text contracts have led to classify rules in *behavioral rules* and *assignation*

rules; the former establishing conditions to the use of some assets. In a behavioral rule, at least the subject and the behavior are identified. An example of this type of rules is provided at the beginning of the Section 4.3: "The Provider will delete the Client's account after the contract termination". This category of rules gets its name from the fact that the object of the rule is a behavior; in other words, these rules control the activities that can be performed where some conditions are fulfilled. On the other hand, assignation rules control the characteristics of the assets by setting some values to a property. These rules are important since they allow to determine the state of an asset.

Behavioral rules require that the subject, behavior and context be verified in the antecedent in order to assign the mode in which the expected behavior is associated to the subject. It verifies that the attributes of the actor exist, that the asset described in the behavior complies with the attribute restrictions, and that the contextual conditions hold to assign a modal (obligation, permission, prohibition, recommendation) to the behavior that the subject has to perform. For example, for the aforementioned rule, "delete" is to be a valid operation for the asset "Client's account" and the state of the contract is verified as part of the context. Consequently, at least both the definition of the subject and behavior are required in the antecedent of a behavioral rule, while in the consequent, the `hasModal` property is required to relate the actor and the activity. Let us recall that the `hasModal` property has as sub-properties `hasObligation`, `hasRecommendation` `hasPermission` and `hasProhibition`.

Formally, a behavioral rule can be defined as a rule having the definition of a modality in the consequent:

$$\begin{aligned} \mathit{BehavioralRule} \subseteq \text{Rule} \wedge \{\mathit{behRule} \mid \exists(\text{behRule}, \text{consqMod}) \in \text{consequent} \\ \wedge\, \text{consqMod} \in \text{ModalConsequent}\} \end{aligned} \tag{4.12}$$

From the contract base, I have identified that the definition of assignation rules is simpler since only conditions about the subject or the behavior are usually verified (rarely about both of them). The definition of the antecedent for the assignation rules is similar to the antecedent of behavioral rules, with the difference that the restriction of having one subject and one behavior was relaxed. In the consequent, the role `hasModal` is replaced by the role `assigns`, which relates a property with its value. The formalization of the `AssignationRule` is presented in Eq. 4.13.

$$\begin{aligned} \mathit{AssignationRule} \subseteq \text{Rule} \wedge \{\mathit{asgRule} \mid \exists(\text{asgRule}, \text{valConsq}) \in \text{consequent} \\ \wedge\, \text{valConsq} \in \text{AssignationConsequent}\} \end{aligned} \tag{4.13}$$

In Section 3.4.1, I have stated that one of the advantages of representing the concepts' attributes as individuals instead of properties is the possibility to assign more complex descriptions to those attributes. Indeed, in assignation rules I am interested in representing that some values for an attribute are not always an asserted fact during the validity of the contract, but that it was

the consequence of the state of the contractual relation at a given time. Such semantics is represented by the relation `assigns`. Thus, the verification that such property be related to an specific concept is part of the conditions verified in the antecedent.

Figure 4.1 presents the structure of the contractual rules. Note that each rule is restricted to having one single subject and one single behavior. This is justified by the accountability needs to uniquely identify the wrong actor and the involved elements in case of faults.

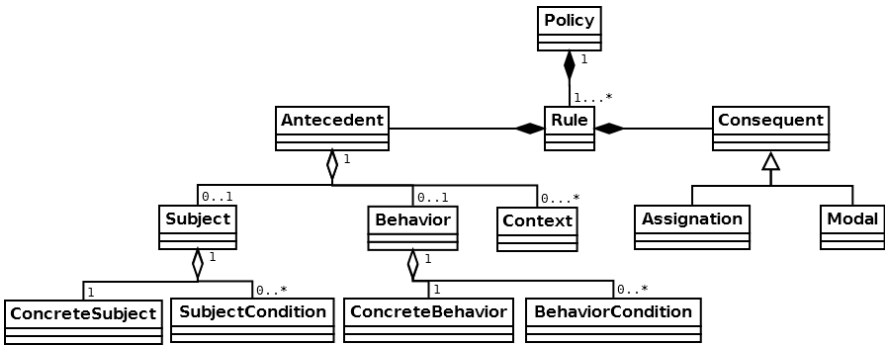


Figure 4.1: Contract Rule Structure

Besides the elements describing the structural part of the rule, some meta-information is associated to the rule definition to completely describe its role in a controllability framework.

4.4.6 Purpose

The purpose represents a justification for the behavior that a subject performs under certain circumstances, i.e, a justification for the rule itself. The main reason for the explicit representation of the purpose comes from its usefulness in the accountability process. In the current approach, the assignation of responsibilities is supported by a semi-automatic reasoning process. It takes the available knowledge about the contract and reasons about *who* is the faulty party and *what* is the affected asset, by finding relations between the different subjects and objects within the contract policy. The purpose adds an additional explanation as to *why* a behavior is requested but from the business point of view. The purpose allows to have a better understanding of the conditions of a fault in order to effectively evaluate the performance of contractual parties. For instance, the justification for an infringement may have been the compliance with some service guarantee or the compliance with other contract rule. Sometimes, organizations prefer to infringe a condition with a low impact than comply with conditions that may infer negative consequences. For instance, let us suppose that a service guarantee establishes the date of the service delivery, while another contract term prohibits to share assets with third parties. If the provider believes that for some reason it wont be possible to deliver the commodity on the agreed date, he/she faces the problem of either to infringe the delivery dates or to subcontract a part of the service to a third party while having to share the client's assets.

Formally, the purpose of the rule is defined as:

$$Rule \subseteq \{r \mid \exists(r, pps) \in \text{hasPurpose} \wedge pps \in \Delta^I\} \tag{4.14}$$

4.4.7 Rule Weight

The weight of a rule, represented in a quantitative form, is used to indicate the degree of importance of that rule from a business perspective. This numerical value is subjective and linked to the relevance of a specific rule for the organization. Therefore, it is associated to the organizational impact of a possible rule infringement. The more important the rule, the heavier the weight. This means, it would be expected that obligations have a more significant weight than recommendations; however, two different obligations can differ in their weights.

In the definition of the `Purpose`, I have affirmed that the infringement of a rule can be due to the decision of not to infringe another rule which may be more important, in terms of either organizational impact (damages) or penalties. Thus, weight helps identify the importance of rules, and it becomes a decisive criterion to evaluate and analyze contract deviations.

The quantitative representation of weights was chosen instead of a qualitative one, because it favors a better distinction between two rules with the same label. That is to say, it allows to overcome the semantic confusion of having to differentiate a recommendation rule which is of "low" importance from an obligation rule with the same level of importance.

In order to assist the contract designer in the assigning of weights, it is advocated that on both clients and providers side, some validation of the weight be done before the contract signature. It allows parties to be aware of possible inconsistencies between modals and weights regarding their inner organizational knowledge. To support the detection of those possible conflicts, the modal scale depicted in Figure 4.2 is mentioned. So, an alert is triggered if the system detects that a recommendation rule about the use of the asset a_1 has a greater weight than an obligation rule which also restricts the use of the same asset. Although such alert is only informative, the goal of the proposed mechanism is twofold. First, it avoids that weights be randomly assigned with no organizational meaning, secondly, it guarantees that when the contract is signed, the involved actor be aware of the implication of the weights in the process of contract and actor assessment. Figure 4.2 shows the suggested modal scale according to the degree of intensity from the weakest modal to the strongest one.

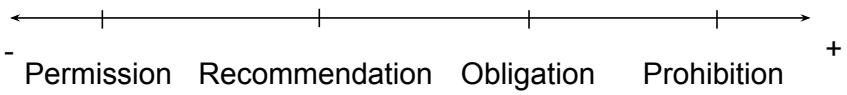


Figure 4.2: Modal scale

4.4.8 Evidence

Traditionally, the compliance with high-level service guarantees[4] has been based on trust, either directly in the external party or in a third party which supports claims made by the external party. In this scenario, each contractual party trusts that his counterparts will act respecting and privileging his business wishes and needs, without performing activities that would cause damages to any service stakeholder. This is of course the ideal situation but not the most common case.

So as to have more concrete information supporting the behavior of the contractual party, specially regarding business commitments[5], first, it is necessary that organizations be able to explicitly represent their needs and claims; secondly, that they have some guarantee which support their accomplishment.

Evidences are guarantees demonstrating compliance with a rule. The importance of the evidence modeling is twofold. First, due to the fact that contract rules define behaviors as abstract activities which can not be monitored in runtime by the information system, evidences take over the runtime verification process. So, the mere existence of an evidence serves to prove compliance with the particular rule it is linked to. Secondly, evidences come to enhance trust by providing concrete elements which support the behavior of a contractual party.

The formalization of the evidence is presented in Figure 4.3. An evidence is defined as anything that guarantees compliance with a rule. That evidence is assumed to be digital in order to be attached to the messages exchanged between the client and the provider via service oriented protocols. In particular, SOAP allows the sending of XML tags as well as binary files, such as images, in the header of messages [65].

An evidence is completely defined by its abstract and concrete parts. The former plays the role of a category to which the concrete evidence belongs to. An example of an evidence is the digital signature of an actor, to which some properties are associated, for example the encryption algorithm, while the concrete evidence is the specific file containing the signature itself. In the abstract part, the property `isRequired` defines the enforceability of the evidence. The properties of the evidence are not restricted to be all automatically verified by an information system. This is because sometimes the human intervention is required to analyze the evidence. Therefore, the property `isComplete` is define to model whether the set of properties automatically verified by an information system is enough to certify the validity of the evidence or whether some analysis of an human expert is needed.

Taking in mind the highly importance of the evidence for the validation process of the contract compliance, I state that the lack of a required evidence is considered as an infringement of a contract term.

It is highlighted that an evidence provides guarantees for any type of rule modality. However, a detailed analysis leads us to conclude that the largest usefulness of evidences is to support obligations and prohibitions. Indeed, as presented in Section 4.4.3, a permission authorizes an actor to carry out an activity under certain circumstances, but in those cases the activities are not mandatory. For most of cases, the verification of the compliance with recommendations and permissions could be done by checking the context. For example,

[4] Guarantees represented in plain-text documents corresponding to business requirements

[5] A business commitment means a high-level contractual rule, which can not be verified in runtime by the information system by monitoring single actions on objects.

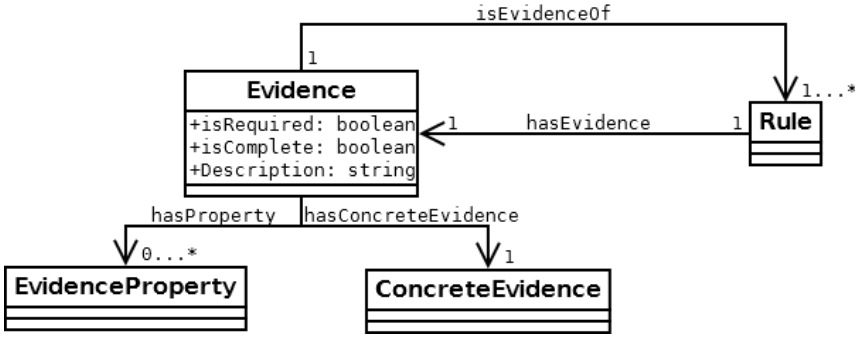


Figure 4.3: Model for the Rule's Evidence

the permission to subcontract a task could be verified by checking if this task is late. From the risk management point of view, having evidences that prove that a service stakeholder has done something that he/she was anyway authorized to do is not as important as proving that he/she did not do a forbidden activity.

As far as security and legal issues are concerned, the collection of evidences is not seen as spying or illegal surveillance because all the data collected during the contract validity is explicitly indicated in the contract, it includes the meta-information forming the evidence, which can include automatically collected data such as IP addresses or locations. In general, I define evidences as meta-information justified by the fact that they are not an explicit part of the service's commodity, but provides additional information about the service provision.

Figure 4.4 illustrates the semantic definition of rules, where the relation `hasState` is the object of study of the next chapter and the relations `isSetBy` and `hasBeneficiary` has been defined in Chapter 3.

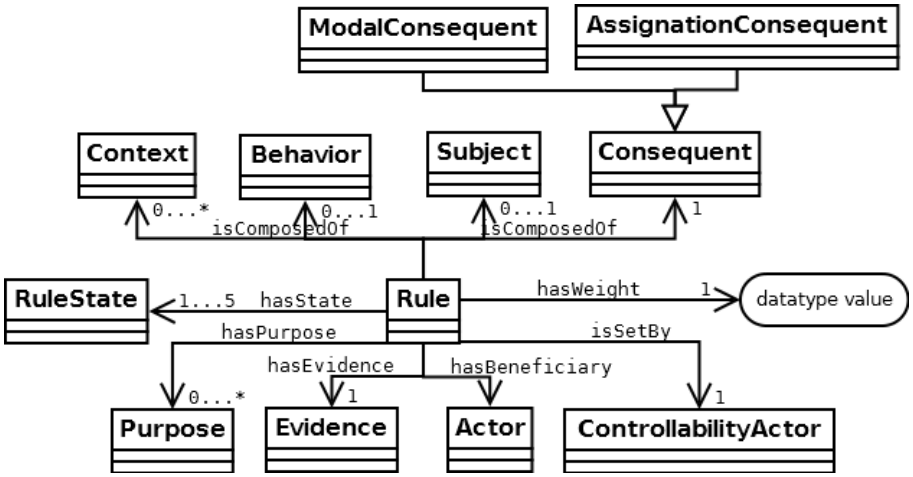


Figure 4.4: Rule-based Model for the Contractual Terms

4.5 Machine-readable Contractual Terms

In the previous section, a model of policies aiming to describe the rules governing the relation between clients and providers was presented. Such rules were classified in behavioral rules, which assign restrictions to the behaviors that actors can perform; and assignation rules, which assign a value to a contractual element based on some conditions associated to an actor, an asset or the context. Similarly, arguing that it is needed to have more than just trust about the policy compliance by the contract parties, the party's behaviors is supported with evidences. I have also included to the rule definition some meta-information, which will be useful for the heuristic analysis of the service provision. In this Section, I will demonstrate the validity of this model through its implementation in a machine readable form.

One of the most important features of SOA is to target the inter-operability between different information systems. In order to do so, SOA relies on standard languages based on XML. I am therefore interested in using a XML-based structure for the machine-readable representation of the contractual policy. Such a representation should be able to support the definition of rules as well as to allow the inclusion of references to the ontology model[6]. I aim to highlight that the contribution of this Chapter is not the proposition of a new language to express business requirements, but a model which formalizes the elements needed to represent such a requirement. Therefore, I have left the decision of the concrete language to use to the developer. However, in this Section, to validate the model some rules are represented in a machine-readable form. Here, instead of creating a new complete rule-based language from scratch, an existing one was reused, and I modify it by including the elements proposed in the model. This allows to take advantage of current representations in terms of parsing, reasoning and validation.

Rule ML Inc. is an organization seeking the standardization of the rule specification. As a result of the increasing adoption of web technologies, Rule ML Inc. aims to integrate its RuleML (Rule Modeling Language) specification as part of the web stack and the semantic web technologies by acting as a bridge for current technologies. It has given rise to languages such as SWRL and SWSL, which are both based in RuleML. The underlying formalism of RuleML relies on a subset of the FOL and the current version of its XML Schema[7] includes several modules representing its sublanguages (and therefore its expressiveness), among them the Datalog module and the Hornlog module. The former one is a subset of the Horn logic (hornlog module) which allows the definition of facts (assert and retract), rules and queries. The latter is also based on a horn logic formalism but adds the possibility to represent functions and complex expressions.

SWRL is one of the Rule ML efforts of rule specification, which allows to integrate OWL constructors to the definition of rules. That is to say, SWRL can be defined as the combination of RuleML and OWL. The SWRL model inherits from RuleML the top-level structure of rules and the definition of variables; and from OWL the definition of the header elements, class elements, property elements and instances to define the particular valid atoms in the *if* and *then* part of the rule. In general, it provides a language to express rules where the

[6] The machine-readable representation of the ontology model was also presented in Section 3.5 in a XML language.

[7] At the moment this thesis was written, it is the version 1.03

semantic of the literals is taken from the ontology. Syntactically, rules are constructed by following a Horn-like form, where some built-ins constructors could be applied, such as greaterThan, equalTo, date, add, pow, among others. Concretely, SWRL formalizes the following atoms:

- Class Atom: It is composed of an OWL constructor (Class, data restriction, object restriction, oneOf, UnionOf) followed by either an individual or a variable.
- Data range atom: It is composed of an OWL constructor (data restriction, object restriction, oneOf, UnionOf) followed by either a variable or a data-typed value.
- Individual property atom: It makes references to an OWL property, followed by two individuals, two variables or any combination of them.
- Data valued property atom: It makes references to an OWL property, followed by either an individual or a variable, and a data-typed value.
- Same individual atom: it is a SWRL constructor which compares if two variables or two individuals are equal.
- Different individual atom: it is a SWRL constructor which compares if two variables or two individuals are different.
- Built-in atom: they are built-in operations provided by SWRL which allows to carry out some operations on boolean values (boolean Not), strings (concat, upper case, lower case, contains, substrings,...), and dates (time, date, add year, duration, subtract year, add day,...)[8].

In general, the expressiveness of SWRL is tied to the ontology, which means that the elements composing a rule should be a class expression, a property expression or some SWRL built-in. Note that although SWRL allows to reuse the semantic model for the definition of contractual rules, its model based on atoms does not allow to make meaningful inferences about the compliance with the contract rules since the description of atoms is not specific enough to give meaning to the elements described in the rule; for instance, it is not possible to distinguish if an actor is described as the obligee or as some assets used in the service provision. However, it has the advantage of its direct integration with OWL, and it is supported by several inference engines such as Pellet and Hermit.

In the machine-readable representation of policies, some of the SWRL definitions are inherited since a subject can be seen as the integration of a class atom with an individual property atom, and a context can be defined as a list of individual property atoms. Indeed, for the purpose of validate the proposed model through a concrete implementation, the syntax of the rule model was implemented on the basis of the XML representation of SWRL and RuleML.

[8] The complete list of built-in is presented in `https://www.w3.org/Submission/SWRL/`.

4.6 Examples

Below, the concrete syntax in XML is illustrated by two examples. The first one extracted from the business agreement of Dropbox[9]. That document written in plain English establishes the terms governing the relation between Dropbox and any organization using and accessing Dropbox's services. The second example is extracted from a contract taken from the Oil & Gas field. Following, the proposed terms are semantically analyzed and a prefix predicate notation is proposed for the knowledge representation. Then, those predicates are taken as the basis for explaining the XML representation.

Example 1. Dropbox Term of Service

> *Some Services allow Customer to download Dropbox software which may update automatically. Customer may use the software only to access the Services. If any component of the software is offered under an open source license, Dropbox will make the license available to Customer.*

By following the methodology described in Chapter 3, let us first to tag the elements described in the above term to subsequently establish the relation among those concepts.

Asset(`Services`)
Actor(`Customer`)
OrganizationalOperation(`Download`)
Application(`Dropbox_Software`)
Attribute(`Updating`)
OrganizationalOperation(`Use`)
OrganizationalOperation(`Access`)
Application(`Software_Component`)
Provider(`Dropbox`)
Asset(`License`)
Attribute(`License_Type`)

Note that `Services` and `Customer` were defined as belonging to the class `Asset` and `Actor` instead of the classes `Service` and `Client`, respectively. It is because they act as roles. That is to say, in the header of the agreement, the concrete services offered by the provider are described and they are tagged as individuals belonging to the class `Service`. Later, each of those individuals is used together with the property `identifies` to assign them the label `Services`. Consequently, such individual is considered as an asset which represents the set of services regulated by the terms of the agreement. Similarly, the `Customer` acts as a label identifying the concrete client.

`Dropbox_Software` and `Software_Component` are tagged as applications (and consequently as an asset).

[9] `https://www.dropbox.com/privacy?view_en#business_agreement`

Finally, note that `updating` is defined as an attribute instead of as an operation. However, in this case, no control is intended to be operated on the updating, instead it represents a feature of the asset defining what the asset can itself do and not what an actor can externally do with it.

After the identification of the individuals belonging to each concept of the model, I analyze each sentence in order to determine the relation among concepts. Thus, the first sentence describes an activity associated to the operation `download` which can be performed on the `Dropbox_Software`, to this latter the property `updating` is associated as part of its definition. This first sentence is formally represented as:

BusinessActivity(`Download_Software`)
employs(`Download_Software`, `Download`)
employs(`Download_Software`, `Dropbox_Software`)
hasProperty(`Dropbox_Software`, `updating`)
hasValue(`Updating`, `automatically`)
employs(`Dropbox_Software`, `Software_Component`)

Note that given the inclusion of the existential quantification, it is not possible to assert that all Dropbox services allow to download a software. However, the knowledge *hasPermission*(`Customer`, `Download_Software`) needs to be verified in order to associate behavioral restrictions to the services which allow the downloading of software. The second sentence expresses a behavioral rule and the purpose of that rule:

BusinessActivity(`Access_Service`)
BusinessActivity(`Use_Software`)
employs(`Access_Service`, `Access`)
employs(`Access_Service`, `Services`)
hasPurpose(`R1`, `Access_Service`)
employs(`Use_Software`, `Use`)
employs(`Use_Software`, `Dropbox_Software`)

`R1` verifies that the customer has the right to download the software, to assert that he/she can use it to access the service. `R1` can be express as:

Actor(`Customer`) $\wedge$ *BusinessActivity*(`Use_Software`) $\wedge$
hasPermission(`Customer`, `Download_Software`)
$\Rightarrow$ *hasPermission*(`Customer`,`Use_Software`)

The third sentence indicates that in the knowledge base Dropbox asserts a new value for the property `available_stakeholder`, i.e. *isSetBy*(`R2`, `Dropbox`), where `R2` is formalized as:

$Actor(\texttt{Customer}) \wedge Asset(\texttt{License})$
$IsEmployedBy(\texttt{License}, \texttt{Software_Component}) \wedge$
$hasAttribute(\texttt{License}, \texttt{License_Type}) \wedge$
$hasLiteralValue(\texttt{License_Type}, \texttt{Open_Source}) \wedge$
$hasAttribute(\texttt{License}, \texttt{available_stakeholder})$
$\Rightarrow assigns(\texttt{available_stakeholder}, \texttt{Customer})$

As it was previously mentioned, the definition of `R1` and `R2` is very similar to the human-readable syntax of SWRL, in the sense that it falls within the definition of attribute atoms and class atoms. It is not surprising since an ontology representation is composed of monadic and dyadic predicates. However, such a representation lacks structure since any concept defined within the semantic model can arbitrary be used to define a rule, which restrains a general formalization of a controllability rule. Consequently, it will be requested to label the role of each of these components by using the proposed model. Note that the definition of atoms in SWRL does not allow to identify the role played by each element of the rule, which makes it difficult to assign a clear semantics to them, make queries about the whole policy, verify their correct structure according to the knowledge to be asserted and make further processing such as assignation of liabilities. For instance, the tagging process of this contract term shows that at least seven concepts can be classified as assets. So, if a rule comprises two or more assets, the SWRL model does not allow to know which asset corresponds to the behavior of the rule (it means, assets being affected by the actor) and which ones are part of the context or the subject (actors are also organizational assets).

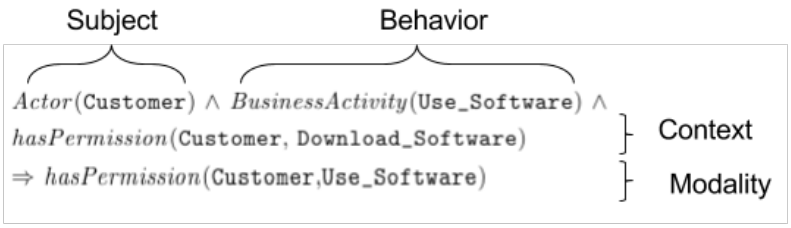


Figure 4.5: Tagging of the Rule's Elements

Concretely, in order to define subject, behavior and context, it is requested to refer to the `owl:Class`, `owl:Individual`, `owl:ObjectProperty` and `owl:DatatypeProperty`. The concrete XML representation of `R1` and `R2` is presented in Listing 4.1. Although SWRL uses the version 0.8 of RuleML, in the following the last stable version, Deliberation RuleML 1.03[10] is proposed.

Listing 4.1: XML Syntax for the Dropbox Term

```
<ruleml:Implies id="R1">
 <ruleml:if>
```

[10] `http://wiki.ruleml.org/index.php/Specification_of_RuleML#XSD`

```
  <pol:Subject>
   <pol:ConcreteSubject owlx:name="&exDropbox;Customer" owlx:type="
       &contract;Actor"/>
  </pol:Subject>

  <pol:Behavior>
   <pol:ConcreteBehavior owlx:name="&exDropbox;Use_Software" owlx:
       type="&contract;BusinessActivity"/>
  </pol:Behavior>

  <pol:ContextList>
    <pol:Context>
     <pol:Property pol:PropertyName="&contract;hasPermission"/>
     <pol:arg1 owlx:name="&exDropbox;Customer"/>
     <pol:arg2 owlx:name="&exDropbox;Download_Software"/>
    </pol:Context>
  </pol:ContextList>

 </ruleml:if>
 <ruleml:then>

  <pol:ModalConsequent>
    <pol:Property pol:PropertyName="&contract;hasPermission"/>
    <pol:arg1 owlx:name="&exDropbox;Customer"/>
    <pol:arg2 owlx:name="&exDropbox;Use_Software"/>
  </pol:ModalConsequent>

 </ruleml:then>
</ruleml:Implies>

<ruleml:Implies id="R2">
 <ruleml:if>

  <pol:Subject>
   <pol:ConcreteSubject owlx:name="&exDropbox;Customer" owlx:type="
       &contract;Actor"/>
  </pol:Subject>

  <pol:Behavior>
   <pol:ConcreteBehavior owlx:name="&exDropbox;License" owlx:type="
       &contract;Asset"/>
   <pol:BehaviorCondition>
    <pol:Property pol:PropertyName="&contract;IsEmployedBy"/>
        <pol:arg1 owlx:name="&exDropbox;License"/>
        <pol:arg2 owlx:name="&exDropbox;Software_Component"/>
   </pol:BehaviorCondition>
   <pol:BehaviorCondition>
    <pol:Property pol:PropertyName="&contract;hasAttribute"/>
        <pol:arg1 owlx:name="&exDropbox;License"/>
    <pol:arg2 owlx:name="&exDropbox;License_Type"/>
   </pol:BehaviorCondition>
   <pol:BehaviorCondition>
    <pol:Property pol:PropertyName="&contract;hasAttribute"/>
        <pol:arg1 owlx:name="&exDropbox;License"/>
    <pol:arg2 owlx:name="&exDropbox;available_stakeholder"/>
   </pol:BehaviorCondition>
  </pol:Behavior>

  <pol:ContextList>
    <pol:Context>
      <pol:Property pol:PropertyName="&contract;hasValue"/>
      <pol:arg1 owlx:name="&exDropbox;License_Type"/>
```

```
      <pol:arg2 owlx:name="&exDropbox;Open_Source"/>
    </pol:Context>
  </pol:ContextList>

 </ruleml:if>
 <ruleml:then>

   <pol:AssignationConsequent>
    <pol:Property pol:PropertyName="&contract;assigns"/>
    <pol:arg1 owlx:name="&exDropbox;available_stakeholder"/>
    <pol:arg2 owlx:name="&exDropbox;Customer"/>
   </pol:AssignationConsequent>

 </ruleml:then>
</ruleml:Implies>
```

By generalizing the previous example, the concrete XML syntax of the proposed approach is defined in Listing 4.2. In the machine-readable representation I intend to extend the *if-then* part of RuleML to include the three pillars of the proposed model, namely: subject, behavior and the list of contexts. I reuse the datatypes proposed in the XML representation of OWL to describe the concrete subject and object. Similarly, the definition of attributes for the subject and object expects two element, i.e. the property and one individual. It differs from the definition of Individual PropertyAtom where three elements are expected, namely: the property and two individuals related by such a property. This was purposefully made to restrict that the property used to define the attributes includes effectively as a domain the actor specified in the concrete subject. Instead of representing the attributes as *property*(`arg1`, `arg2`), only the `arg2` is explicitly required, while the `arg1` is implicitly supposed to be the individual defined in the attribute `owlx:name` of the concrete element.

Note that this model does not include variables in the definition of rules. Indeed, from the analysis of the base of contracts, when a particular term makes generalizations to refer to all the elements of a given class, it is done by using a role previously defined in the header of the contract, which in the model is represented by means of an individual. For instance, the individual `Services` in the Dropbox example.

Listing 4.2: XML Concrete Contract Rule Syntax

```
<ruleml:Implies id=(xsd:anyURI)>
 <ruleml:if>
    (pol:SubjectGroup | pol:BehaviorGroup | pol:ContextGroup)
 </ruleml:if>
 <ruleml:then>
    (<pol:AssignationConsequent> | <pol:ModalConsequent>)
 </ruleml:then>

SubjectGroup:= <pol:Subject>
                 (pol:ConcreteSubjGroup, pol:ConditionSubjGroup1*)
               </pol:Subject>

ConcreteSubjGroup:= <pol:ConcreteSubject
                      (type=owlx:individualIDAttrType)/>

ConditionSubjGroup1:= <pol:SubjectCondition>
                        (pol:PropertyGroup, pol:argumentGroup2)
                      </pol:SubjectCondition>
```

```
BehaviorGroup:= <pol:Behavior>
                  (pol:ConcreteBehGroup, pol:ConditionBehGroup1*)
                </pol:Behavior>

ConcreteBehGroup:= <pol:ConcreteBehavior
                     (type=owlx:individualIDAttrType)/>

ConditionBehGroup1:= <pol:BehaviorCondition>
                       (pol:PropertyGroup, , pol:argumentGroup1,
                       pol:argumentGroup2)
                     </pol:BehaviorCondition>

ContextGroup:= <pol:ContextList>
                 (pol:PropertyGroup, pol:argumentGroup1,
                  pol:argumentGroup2)
               </pol:ContextList>

PropertyGroup:= <pol:Property
                  pol:PropertyName(owlx:PropertyName)/>

argumentGroup1:= <pol:arg1 (type=owlx:individualIDAttrType)>

argumentGroup2:= <pol:arg2 (type=pol:DataIndividualType)>

DataIndividualType:=(owlx:datatypeAttrType |
                     owlx:individualIDAttrType)

AssignationConsequent:= (pol:PropertyGroup, pol:argumentGroup1,
                        pol:argumentGroup2)

ModalConsequent:= (pol:PropertyGroup, pol:argumentGroup1,
                   pol:argumentGroup2)
```

In general, in comparison to SWRL, the proposed model gives a more significant meaning to the elements described in the rule by replacing the definition of atoms in the structure of the rule, by the subject and its attribute, the behavior and its attributes, the context and the consequent. However, unlike SWRL, the use of variables was omitted and object properties followed by one single argument are allowed. Consequently, the `arg1` of the proposed model is restricted to be an individual, while `arg2` can either be an individual or a data-typed value. Indeed, it corresponds to the definition of `RelationalAttributes` and `QualityAttributes`.

For the implementation of the proposed model a XML schema was developed, which inherits both the current stable version of the datalog module of RuleML and the XML schema of SWRL. In order to execute contractual rules, a RuleML engine is needed as well as the modification of its inherent parser, which allows to correctly process the added XML tags. The inclusion of a `type` attribute in the definition of rules will allow to link their syntactic description in XML to their semantic meaning, by using the URI defined in the semantic model to attach some metadata to the rule definition.

Example 2. Oil & Gas Term of Service

> *Operators are restricted to be in the plant for sending project progress data to PPMS.*

This example comes from the Oil & Gas contract presented in Figure 3.3. For

this example the tagging process is not presented, but it can be inferred by the references to the OWL classes. That contract term was selected since although it is short in the explicit knowledge represented, it also allows to illustrate some of the arguments presented throughout this chapter.

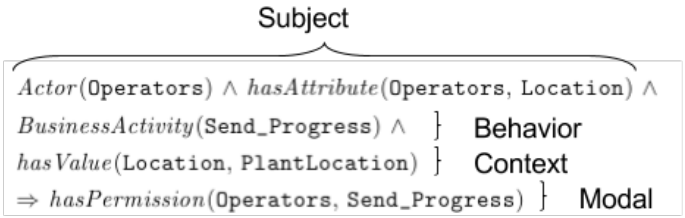
Let us recall that, as described in the contract header tagging in Section 3.5, this contract is signed between the organization PPMS and the OilGasCompany. Therefore, the first finding is that the `Operators` are third parties. Indeed, OilGasCompany requests PPMS the monitoring and management of their projects; also, they subcontract the maintenance, interventions and repairs of his oil plant to an external operator. Consequently, the service provision is composed of three independent organizations interacting between them. Although a contract signed between the operators and PPMS may exist, following I will focus on the contract signed between PPMS and OilGasCompany.

In this contract, it is described that in order to carry out the monitoring process of projects, PPMS requires that operators send them some data regarding the progress of their tasks. A second finding is that no description about the way in which that data should be sent is provided in the contract. That is to say, progress data can be sent by email or by accessing to some information system. If it were done by means of an information system, an access policy at the PPMS side can effectively trace the location of the operator to verify the conditions for the data entry. However, as it was argued in this approach, no knowledge about the way in which the external partner performs his/her activities is available. Supposing that it is done by allowing the access of a third party to its information system, the client (OilGasCompany) can not impose restrictions to the way in which the information system of the provider (PPMS) is internally used; without mentioning that operators are not part of the OilGasCompany, therefore, they cannot be held responsible for the behavior of the operators. All of this holds the previously argument that only the subjects of rules are actors since they are liable of the externally observable activities performed on assets. Similarly, this confirms the assumption that partners are considered as black boxes whose internal processes are unknown.

Despite the fact that OilGasCompany cannot control the internal process of PPMS, it is the controller of the data collected by the operators and processed by PPMS. Therefore, the client is interested in ensuring that such a data is as much factual as possible, which is reflected in the contract as the need to send the progress data after each intervention being on site, such that PPMS can make an effective monitoring of projects and guarantee that operators do not do the job at the last minute. From the semantic point of view, this relation between actors is modeled by using the properties `hasBeneficiary` and `isSetBy` associated to the definition of the rule.

The previous term represented as a rule in a predicate-based notation is expressed as:

Since the location is verified in the antecedent part of the rule, the knowledge about the permission to send the data about the progress of the project is only asserted if the operator satisfies the attribute restrictions. Moreover, it should be highlighted that an obligation modal is not used since it would commit the operators to send data of progress every time the operator is in the plant, which is not the expressed knowledge in the rule. It is only expressed if the



party complies with the contract, OilGasCompany assumes that the operator was in the plant when he/she sent the data progress, which will be verified by comparing the evidences sent by the external partners to OilGasCompany. Listing 4.3 presents the way how the Oil&GasContract can be represented in XML.

Listing 4.3: OWL Contract Rule for the Oil&Gas Example

```
<ruleml:Implies id="http://semanticContract/exampleOilGas/R3">

 <ruleml:if>

  <pol:Subject>
   <pol:ConcreteSubject owlx:name="&exOilGas;Operator"
                        owlx:type="&contract;Actor"/>
    <pol:SubjectCondition>
    <pol:Property pol:PropertyName="&contract;hasAttribute"/>
    <pol:arg2 owlx:name="&exOilGas;Location"/>
   </pol:SubjectCondition>
  </pol:Subject>

  <pol:Behavior>
   <pol:ConcreteBehavior owlx:name="&exOilGas;Send_Progress"
                         owlx:type="&contract;BusinessActivity"/>
  </pol:Behavior>

  <pol:ContextList>
    <pol:Context>
      <pol:Property pol:PropertyName="&contract;hasValue"/>
      <pol:arg1 owlx:name="&exOilGas;Location"/>
      <pol:arg2 owlx:name="&exOilGas;PlantLocation"/>
    </pol:Context>
  </pol:ContextList>
 </ruleml:if>

 <ruleml:then>

   <pol:ModalConsequent>
    <pol:Property pol:PropertyName="&contract;hasPermission"/>
    <pol:arg1 owlx:name="&exOilGas;Operator"/>
    <pol:arg2 owlx:name="&exOilGas;Send_Progress"/>
   </pol:ModalConsequent>

 </ruleml:then>

</ruleml:Implies>
```

To that rule the following semantic meta-information is associated

Listing 4.4: Example of Rule Meta-information

```
<contractTerm rdf:about="http://semanticContract/exampleOilGas/R3">
   <isSetBy rdf:resource="&exOilGas;OilGasCompany"/>
   <hasBeneficiary rdf:resource="&exOilGas;PPMS"/>
   <hasEvicence rdf:resource="&exOilGas;IPLocation">
   <hasWeight rdf:datatype="&xsd;float">0.7</hasWeight>
</contractTerm>

<Evidence rdf:ID="&exOilGas;IPLocation">
   <isRequired rdf:datatype="&xsd;boolean">true</isRequired>
   <isComplete rdf:datatype="&xsd;boolean">true</isComplete>
   <Description rdf:datatype="&xsd;string">
      This evidence collects the IP of the operators at the moment
         he sends the data to locate his position
   </Description>
   <isEvidenceOf rdf:resource="http://semanticContract/
      exampleOilGas/R3"/>
   <hasProperty rdf:datatype="&xsd;string">IP</hasProperty>
   <hasProperty rdf:datatype="&xsd;string">Country</hasProperty>
   <hasProperty rdf:datatype="&xsd;string">Latitude</hasProperty>
   <hasProperty rdf:datatype="&xsd;string">Longitude</hasProperty>
   <hasProperty rdf:datatype="&xsd;string">ISP</hasProperty>
   <hasProperty rdf:datatype="&xsd;date">Date</hasProperty>
</Evidence>
```

4.7 Discussion

The main motivation to carry out this dissertation has been the identification of needs for methods which allow organizations to keep control on their resources when they are not in a domain of protection. Having this need in mind, I reviewed the existing solutions proposed to govern the relation between providers and clients, and I chose to extend the expressiveness of SLA by creating service contracts, which are able to represent more general business requirements including the business expectations of each service actor on the use of his/her resources. In order to express those requirements with a semantic meaning a DL-based model of controllability, framed in a service contract perspective, has been proposed in Chapter 3. Such a model not only clarifies the meaning of concepts such as asset, use, service, controller and processor, but it also allows to identify in the light of those concepts the structure of contractual terms.

Once contract and controllability concepts have been identified, the definition of the contractual terms remained. The analysis of real service contracts, lead us to conclude that contractual terms can be represented as rules since expected behaviors are rarely expressed as assertions, but conditions need to be verified before those behaviors be performed. However, such a complex knowledge cannot be capture by means of the DL formalism. To model this kind of business knowledge, a rule-based formalism was proposed and several approaches were analyzed to determine if any current rule model could be reused to tackle the representation needs. From the literature review, it was concluded that although individually, each model has advantages, no current approach is able to completely represent service contract terms in regard to their semantics, expressiveness, inference and query.

A model combining a significant meaning of the model's entities with a well established formalization for the representation of business rules is set forth. The

former criterion is required in semantics to both define what the rule describes and make meaningful queries, while the latter ensures decidability, inference and a XML-based format. Regarding its machine-readable representation, this model, like SWRL, merges RuleML and OWL, and it overcomes the lack of an inner rule structure and clear meaning of atoms in the definition of business policies, while keeping the monadic and dyadic arity of predicates, as restricted by the OWL representation. Although rules are written in XML they can be executed and inferences can be made since they rely on the Horn logic. Consequently, when this representation is applied to service contracts, it allows, to gain the greatest benefit of both the XML-based syntax aiming at interoperability as one of the principles of the service-oriented architectures and the ontology-based representation aiming at a common understanding of the concepts that are represented in the components of the model. On the other hand, it also has the advantage of a declarative representation such as the non-static definition of the control flow and the rule triggering chaining considering the whole of the rule base.

Regarding the rule model components, some of the design decisions presented in Chapter 3 become clearer, notably the definition of attributes as a concept instead of a role. In order to argue this representation based on the rule model, let us suppose that a particular contract term specifies a property which only exists when certain conditions are met. These categories of terms are found for instance in the cloud, where under some conditions, a paid account can exhibit attributes that free accounts might not have. If in the semantic model, properties have been defined as a role, the new property would not be asserted in the knowledge base since either {`newProperty`(concept, value)} or {`Assignment`(`newProperty`(concept, value)} are not valid constructions for the rule consequent. However, in the proposed model, because properties are defined as individuals, a consequent in the form {`hasAttribute`(<concept>, <`newProperty`>)} is valid since the `hasAttribute` role already exists in the semantic model and only new individuals are going to be asserted. Regarding the two previous (invalid) constructions, on the one hand, the semantics of the assignation rule completely disappears in the first representation and consequently there is no distinction between mutable and immutable attributes. On the other hand, it was supposed for the first representation that the rule model needs to change in a consequent way, thus `Assignment` is proposed as a class replacing the proposed role `assign`. Similarly, a further analysis of this representation shows that its underlying semantic reflects the addition of some knowledge (the `Assignment` class) to the relation {`newProperty`(concept, value)}. Although it is not a valid representation regarding the FOL formalism, the need to express this kind of knowledge has been repeatedly reported in the literature [102][83][44][118], to which approaches such as reification, aggregation and generalization have been proposed. From the technical point of view, the use of reified statements in rules require careful attention in their implementation due to the restrictions of some languages to include anonymous individuals in its rule constructions. However, in the proposed formalization of rules those considerations were avoided as a consequence of the modelling process canvassed for contractual concepts. In short, the proposition presented in this Chapter supports and is coherent with the design choices made for the construction of the semantic model. In addition, it avoids some issues such as the modeling of n-ary relations via reification, without compromising expressiveness

nor creating unnecessary relations or meaningless concepts.

Another contribution of the proposed model is the inclusion of meta-information to the rule definition. It represents data which is not part of (it is not explicitly described in) the contract term but is relevant within the organization to completely define the rule, particularly in terms of controllability. First, contract terms have evidences. Unlike access control, this proposition is oriented towards the definition of coarse-grained activities, instead of atomic and monitorable actions. Indeed, those activities define the behavior of actors, and plain-text contracts are the most visible manifestation that restriction on the actor's behavior are needed in order to avoid organizational damages. However, the challenge is not to semantically define behaviors but to enforce the policy knowing the inherent level of abstraction within the definition of a behavior, and that monitoring at the information system level is not possible since assets comprise a large definition of resources (not only data) and those activities are not always observable at the information system level in the domain of the processor. In this model, the use of evidences has been proposed to support the behavior of actors. Those machine-readable evidences can be sent together with the result of the service, in the case that this one be provided as part of a SOAP message[11], otherwise in specific messages sent from the processor to the controller. As a consequence, the compliance of the actor with the contract policy cannot be determined before the gathering of the evidences, which confirms the usefulness of this approach for a posteriori (empirical) processes such as accountability and service evaluation.

The Directive 95/46/EC [34] highlights the importance of proofs by stating "if the controller fails to respect the rights of data subjects, national legislation must provide for a juridical remedy; whereas any damage which a person may suffer as a result of unlawful processing must be compensated for the controller, who may be exempted from liability if he/she proves that he/she is not responsible for the damage [...]". That is to say, if some damage is claimed by an actor, the defendant can say *it is not my responsibility, and I have the evidence which demonstrate it.* Although I am perfectly aware that the external partner can lie or overestimate his behavior, it wont be a feasible solution to verify each individual's evidence, especially when a large number of external partners are integrated to the business logics. Instead, this approach aims at a notion of semi-trustworthiness. On the one hand, we do not trust partners, so we ask evidence that justifies their behaviors. On the other hand, we assume that partners act in good faith and they do their best to accomplish contractual rules, so initially the provided evidence is trusted.

Second, rules have weights. The weight is a metadata associated to the contractual term to explicitly establish the organizational importance of the contractual commitment described in the rule. Those weights apply to behavioral rules as well as assignation rules. Some works such as [99] have already proposed the use of weighted rules for the definition of inter-organizational policies, but in that case its value is mutable and the weight in directly related to the modal. Hence, they propose that the set of obligation rules has a weight value of 1, interdiction rules of 0, permission rules of 0.5, and thus attempts to violate change the modal of the rule (obligations become prohibitions) and

[11] Let us recall that the use of SOAP messages is favored since they permit the attachment of metadata in the header of the message.

consequently its weight. On the other hand, in my approach the rule's weight is considered invariable and is used to set the degree of relevance of a rule, in a way that it helps determine if the impact of the non-accomplishment of an obligation is more important than the impact of ignoring a recommendation or violate a prohibition. I state that the fact that this weight be explicitly indicated in the rule definition, and consequently in the contract, does not compromise the controller. On the one hand, in plain-text contracts, it is common to find terms which establish the consequences of the non-accomplishment of a contract term. That knowledge is equivalent to the rule weight in the sense that it can be used to evaluate the importance of that particular term. In this approach, for computations reasons I aimed at the definition of this knowledge in a quantitative way. On the other hand, this information is also useful to the processor in order to evaluate the consequences of the non-compliance of his commitments.

Finally, when referring to the provision of a service, it is common to face cases in which more than two independent organizations collaborate to accomplish a required business process. These workflows[12] are highly important in the accountability process due to the responsibility chaining resulted of the complex interactions between organizations. During the analysis of the contracts, I saw that most of the contract terms fall in either behavioral and assignations rules, however a small group represented the fact that under some conditions, an actor delegated activities to a third party. This delegation has a significant impact in terms of liability and accountability. Although the scope of this dissertation does not cover administrations rules (delegation rules), I strive to give some insights to the way in which those terms can be formalized in order to completely define the set of contractual terms.

From the point of view of service contracting, the delegation is understood as to entrust something (a process, a commitment) to another party. In this context, it is not only important to know to whom the policy should be delegated, but more importantly for the accountability process, to establish liabilities in case of some damages were produced as a result of this collaborative relation. Note that up to now, I have restricted the model to have one single client (C_s) and one single provider (P_s); in case of delegation, a new organization ($SP_{s'}$) is considered, which becomes a provider for P_s, but who plays the role of third party in the client's eyes. I argue that those subcontractors have to behave and use the assets according to the governing rules established by the controller; all of this, without exposing the internal business logic of neither of the two organizations, since on the one hand the provider is not always interested in disclosing information about its business partners (subcontractors) and on the other hand, the subcontractors may ignore the existence of the client. Clearly, I am face my initial problem: an organization (provider) wants another external organizational (subcontractor) to use assets according to some rules (established by the client). Therefore, given the cardinality restriction in the contractual parties[13], if new organizations come to be part of the service provision, i.e, third parties, an additional contract should be established, where a subset of the initial

[12] In the literature, terms such as virtual organizations or digital business ecosystems are often used to refer to this inter-organizational relation. Here the term workflow is used in its general meaning to refer to a sequence of processes leading to the provision of a [composed] service to a client.

[13] Let us recall that it leaves the multi-party contract negotiation out of the scope of this work.

contract needs to be propagated to the new party, more concretely, those rules involving the delegated assets.

Although the delegation of a process may be executed by either the client or the provider, the previous analysis focused on a delegation done by the provider. It is because most of the activities which are prone to be delegated come from this side of the interaction. Indeed, the principle behind the orchestration, and in general, behind the composed services, is to link up services in order to provide more complex business functionalities which would not have been possible to offer by a single provider.

In order to formalize delegation, let $P = \{r_1, ..., r_m\}$ be a contractual policy binding the provider P_s and the client C_s. Let $A = \{a_1, ..., a_j\}$ be the set of assets defined in the behaviors of P, with $j \leq m$; and let BP be a process to be delegated by the provider.

Following the principle of orchestration, the provider knows both his/her partners and the specification of his/her own business process. Thus, let A_{BP} be the set of assets used in BP. The delegated policy $P' = \{r'_1, ..., r'_k\}$ is such that

$$A' = A_{BP} \cap A \tag{4.15}$$

$$\{\forall a' \in A', \exists bh' \mid bh' \in r' \wedge a' \in bh'\} \tag{4.16}$$

where $A' = \{a'_1, ..., a'_k\}$ is the set of assets defined in the behaviors Bh' of P' and $k < j$. It means that the behaviors in the delegated policy correspond to the activities in which the assets used by the delegated process BP are involved. Consequently, all the policy is not shared but only the rules restricting the behaviors on the delegated assets.

Due to the fact that $A' \subseteq A$ holds, Eq. 4.17 also holds.

$$P' \subseteq P \tag{4.17}$$

Finally, the delegation should also consider a change of role for the provider, as depicted in Figure 4.6. Indeed, for the $contract_{ij}$ the organization Org_2 keeps its role of provider, while for the $contract_{kz}$, this same organization becomes the client of the Org_3, due to the delegation process. Therefore, I added the predicate *switch* which reflect the fact that a provider becomes a client (Eq. 4.18).

$$delegates(P', SP_{s'}) \rightarrow switch(P_s, SP'_{s'}) \tag{4.18}$$

4.8 Conclusion

Just a few decades ago, when someone was committed to providing a service, one's his word was enough to believe that the provider will act in good faith in regarding to the provision of the service according to the client needs and expectations. Nowadays, this is far from being the most common case. Technology has changed the way of doing business, and consequently, the relation between clients and providers. The architecture oriented to services has favored the opening of an organization to the worldwide market by facilitating integration and communication with other partners regardless the technological

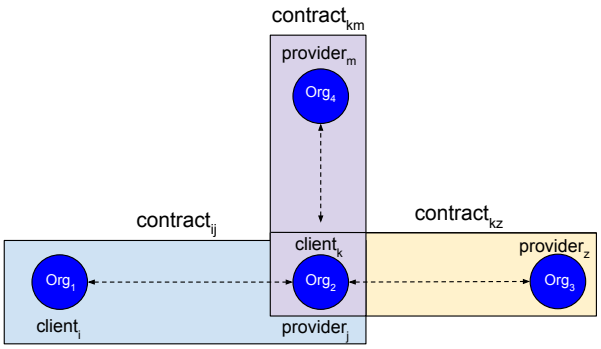


Figure 4.6: Service Workflows and Contracts

heterogeneity or geographic distance. Such a paradigm has not only addressed the needs of dynamism but has also increased competition among organizations.

In such a context where the service provision is mediated by the information systems, several challenges arise regarding the interaction of partners. In particular, due to the fact that even if organizations collaborate during the execution of some business process, they can have opposed objectives, and the behaviors performed by the client can affect the provider and vice versa. This situation is exacerbated by considering that partners can be freely added to the service provision chain, where each partner has no or almost none guarantee about whether each external organization is going to behave "correctly".

Currently, Service Level Agreements (SLA) in both plain text and machine-readable formats are proposed to explicitly describe service guarantees and restrictions governing the service provision. However, the expressiveness of the natural language allows to represent knowledge with a rich semantics. In contrast, machine-readable SLA has a limited expressiveness to represent conditions governing the service provision. Supported in the findings from the literature review, I have concluded that current approaches describing service guarantees and policies governing the service provision lack a clear semantics, define a limited set of actions and objects, and are not able to represent complex knowledge.

In this Chapter, I have presented a model able to represent controllability rules. This proposition does not aim to replace existing security models but to overcome the lack of formalizations to represent business requirements about the expected behavior of the service actors, specially, regarding the use of assets. Thus, this contribution complements existing approaches which focus on fine-grained actions and performance guarantees verified in runtime, with a model which allows to represent coarse-grained activities associated to the business logics of each organization. Concretely, a controllability rule represents the use of assets by defining three main elements, namely: a subject, the expected behavior and some contextual conditions. Moreover, some metadata is attached to the semantic meaning of rules to define the organizational importance of the guarantee (weight), its purpose and the evidence which support their compliance.

The contribution of the proposed model of controllability rules is twofold. First, to be able to represent service guarantees on a more wide range of organi-

zational assets and activities, both based on formal semantics; secondly, to fill the gap between the business and technical perspective of a service provision. Still, two main issues are left as part of the future work, On the one hand, the implementation of the proposed model in an OWL syntax. On the other hand, to improve the formalization of controllability rules to favor alternative behaviors. Indeed, with the proposed model is not directly possible to represent the knowledge that an actor must (obligation) perform an activity A_1 or an activity A_2. Although the weight of the rule can be used to represent part of this knowledge, there is no way to know that the accomplishment of a rule is equivalent to comply with another rule. Consequently, the modeling of an "else" block is part of the future work.

Part III

The Process of Asset Controllability

Chapter 5

Knowledge-driven Reasoning on Controllability

Having tools that to a greater or lesser degree are able to express business requirements for a service provision, is important to clarify the terms governing the relation between the actors of the service. However, explicitly establishing those terms is not enough to guarantee their compliance. In the previous Chapter, a model aiming at the definition of controllability rules as part of the contractual terms was proposed. In this Chapter, I will focus in the process part of the controllability governance by aiming at the assessment of the conformity to the contract regarding the compliance with the established rules.

Objectives

- To verify the compliance with the controllability rules in order to evaluate the performance of the service actors and the quality of the provided service.

5.1 Introduction

The life-cycle of any agreement comprises several stages, namely, authoring, negotiation, approval, execution, analysis, and termination. Machine-readable SLAs, as a particular case of agreements, are consistent with those stages by proposing a life-cycle covering the agreement request (provider discovery, creation of the SLA template and negotiation), operationalization (execution and monitoring), and removal (renewal or termination). In the operationalization stage, the monitoring process aims at the compliance with the agreed guarantees and Service Level Objectives (SLO)s. In Chapter 3, it has been shown that current machine-readable agreements mainly cover security, dependability,

monetary terms and performance aspects as part of the non-functional requirements. Although several techniques have been proposed to prove compliance with those kind of requirements, they coincide in having a runtime process matching the expected parameter value and the actual monitored value of the parameter. According to Wu et al. [114] an un-fulfillment is defined by the Principles of the European Contract Law as corresponding to three cases: first, a defective performance, which means a gap between the monitored data and the expected one; secondly, late performance, which covers a "service delivered at the appropriate level but with unjustified delays"; and finally, no performance covering the commitments which are not accomplished at all. The fact that an assumption based on runtime monitoring restricts the definition of the service level objectives has the advantage of a reactive decision about the path to take in case of faults based on factual data collected during the provision of the service. However, it also limits the set of commitments governing the relation between clients and providers to observable actions and measurable parameters.

In the previous chapters, a model able to represent guarantees about the use of assets was presented. The proposed model of controllability rules is able to extend the expressiveness of the traditional parameter-datatyped value guarantees, by including a formalized semantics to the elements describing the subject, the expected behavior and contextual conditions. However, the price to be paid to extending the expressiveness of the service agreements is to represent activities and attributes which cannot be monitorable or measurable in runtime, which hardens the evaluation of their compliance through traditional monitoring techniques. Moreover, although contracts have a XML representation, they have legal value and can be used to claim the misbehavior of a contractual party. Nevertheless, using contracts only in case of faults neither allows to know what really happened during the service provision nor leverages the strengths of the knowledge that can be collected during the contract execution. These two identified needs led us to propose an enhancement of the traditional compliance with service agreements based on runtime monitoring techniques, with an *a posteriori* process which semi-automatically verifies the compliance with the service contract and support some organizational processes such as the actor assessment based on the expected behaviors and the selection of the provider.

This Chapter is organized as follows. In Section 5.2, I will present the state of the art which clarifies the contribution regarding the existing works proposed in the literature. In Section 5.3, I will present a semi-automatic auditing process to verify the compliance with the contract which is grounded in the two previous proposed models. This process is subsequently illustrated with an example coming from the Oil & Gas industry. In this example the rules representing the contractual terms are presented as well as the step by step process leading to the evaluation of the contract, where different cases are performed to test the process' outcome face to different policy-related situations which may occur during the provision of the service. Later, the available knowledge about the state of the controllability rules is analyzed in order to draw useful organizational conclusions about the behavior of the external partner and to give feedback to the policy. Finally, the conclusion is presented in Section 5.5.

5.2 State of the Art

Following, I will present the state of the art in the fields covering the main contributions of this part of the dissertation. Reviewed works have been grouped according to the process addressed during the agreement life-cycle. Concretely, the literature review focused on works proposing methods to enforce the compliance with some policies and claims about the service provision. Similarly, I have been interested in analyzing current techniques used to perform auditing and assignation of responsibilities in case of faults.

Claim and Policy Enforcement

The enforcement process refers to the process of verification that actors comply with their promises (claims) and that, in general, the policies governing the service provision are respected. Since different kinds of terms[1] can govern the service and its involved actors, several enforcement techniques have been proposed according to the nature of the term and the way in which it is represented.

On the one hand, the literature review focuses on monitoring, which has been the most frequently technique proposed to prove compliance with a service agreement. Monitoring is based on a matching process between runtime collected or derived data with the expected value of pre-established metrics taken from the SLA [1]. It is often done by enabling a trusted third party to take measures at the provider side and inform the parties in case of faults.

In [32], an architecture to detect SLA violations in cloud infrastructures is suggested. The authors argue that monitoring techniques depend on the resource. Therefore, monitoring approaches used in grids cannot be applied to the Cloud. This work aims to map low-level monitored metrics to high-level SLA parameters, as well as to detect a threshold to predict if a monitored value may lead to a violation. Similarly, more recent works [51] propose the prediction of the violation based on usage profile history. In [41] an Eclipse workbench to report the monitorability of services for SLA is described. This work proposes the creation of a log containing in its XML structure a description about the type of each step of the assessment, its result, an optional selection item (feature/monitor), the location for the SLA processes item and a monitoring configuration.

Another work proposing a monitoring architecture is presented in [19]. The goal of this approach is to aggregate and integrate performance data collected in real-time to adapt the resource management workload changes. In [42] an approach based on genetics algorithms is proposed to optimize the recomposition required in cloud environments after a violation is detected by a real-time tracing process. Having in mind the lack of human-readable models, and the coupling between the monitoring configuration and the SLA, in [78] it is proposed a platform able to gain in expressiveness for the SLA metrics and give explanations about the detected violations.

In the previous chapters, I have argued that the definition of the SLO in terms of parameters such as availability and response time seem to be appropriate, and even suitable, for cloud services, because in such a case there is

[1] Here, a "term" is used in its broad sense as any explicit restriction or fact regulating the service or the contractual parties.

an alignment between the technical and the business view of a service. Those parameters correspond to actual attributes of the commodity. Therefore, the first finding about the SLA enforcement based on monitoring is that although agreements apply to service-oriented environments on the whole, current works mainly focus on the cloud. Indeed, the results of a research on specialized journals[2] show that a 91,1% of the works on service level agreement monitoring between 2013 and 2016 address the compliance with Cloud or infrastructure low-level parameters. Similar to the way monitoring techniques of grid cannot be applied to the cloud [32], cloud monitoring techniques cannot be directly applied to SOA, where other kinds of commodities are considered; moreover the quality of the service cannot only be expressed in terms of performance parameters. As described in the survey presented in [1], the monitoring in the Cloud focuses on computation-based and network-based metrics, where the monitoring goals include billing, performance, security, capacity and resource management. Consequently, more efforts should be made to guarantee that other non-functional requirements, such as business wishes about the use of assets, are fulfilled, which are equally important to evaluate the quality of the provided service. Secondly, monitoring is not an isolated task; it can be used to feed other processes including the monitoring process itself. Basically, results of the monitoring can be used to predict future violations, to adapt the service composition, calculate penalties and accounts, where logs are the most common implementation to save collected data.

On the other hand, in [10] [7] [6] [22] [8] [9] not only performance parameters but also security and dependability properties are considered for both the Cloud and SOA. Unlike the aforementioned approaches, those non-functional property categories are guaranteed by certificates, whose enforcement is proposed as a two-step process. In the offline step, a certification authority delivers a certificate to a service model based on tests made during the production phase. In the online step, an evaluation body constantly verifies the validity of the certificate able to revoke it or renewed it with lower guarantees, if the actual value of some property does not meet the expected conditions. In this approach, although the real-time values are obtained via a monitoring process, the notion of evidence is introduced to represent the results of tests. Basically, test evidence is attached to the certificate in the form of metadata in order to provide information useful for the selection of services based on certified properties. Such selection can be done either by a comparison and matching process or by following an e-auction strategy.

Although the enforcement of certified properties is based on formal models and empirical information represented as evidences, the set of concrete non-functional properties needs to be observable in runtime through its attributes. Particularly, those attributes represent their implementation mechanisms. For instance, the confidentiality concrete property belonging to the security category is certified according to the value of the key length, the implemented encryption algorithm and the used protocols.

Other enforcement approaches have been proposed for SOA to verify the compliance with security properties. In particular, for the WS-* family of policies, an architecture composed of a Policy Enforcement Point (PEP), a Policy Decision Point (PDP), and a Policy Information Point (PIP) are set out. The

[2] The research was focused on IEEE and Springer journals

PEP intercepts the attempts of accessing some resource and requests the PDP about the user's rights. The PDP analyzes the request to grant an authorization by evaluating the policy. The result of this process is sent back to the PEP which is responsible for enforcing the decision. If during this process, the PDP needs additional information about the context and attributes, the PIP is in charge of providing this data to the PDP. This architecture has been implemented in the XACML standard and most of the access policies, including usage control policies (the UCON family). Similar to the previous works, this architecture aims at a real-time and automatic detection of the policy compliance.

In conclusion, the principle behind certificates is similar to the approach presented here when considering that in a service-oriented paradigm, the external partner cannot be controlled, and may tend to hide misbehavior, which highlights the need of having evidences. Unlike my approach, certificates aim to guarantee concrete security, dependability and performance properties, which are defined by the value assigned to their attributes. Such a definition highlights that its enforcement mechanism cannot be directly applied to verify compliance with the use of assets. Indeed, I consider a larger set of asset, coarse-grained activities, and abstract parameters which cannot be always verified in runtime. Therefore, in the best-case scenario, test cases can be implemented to validate compliance with only a subset of guarantees reflecting attribute qualities, and a subset of contractual terms reflecting assignation rules. An unsolved problem about whether evidences can be used to support compliance with behavioral rules, and what method to implement to overcome the need of a runtime enforcement. The latter also becomes the main limitation to implement SLA monitoring techniques to controllability policies. Basically, SLA models associate each parameter used in the definition of guarantees with a measurable metric whose value is monitored in runtime. Finally, although the incorporation of a Trusted-Third Party (TTP) can be a solution to enforce behavioral rules, considering the compliance with a contract only in terms of a boolean result reported by the TTP does not allow to exploit the usefulness of the knowledge represented in the proposed models.

Accountability

Due to the economic and social implications of the collaborative service provision, the assignation of responsibilities has increasingly received attention. Beyond legal aspects regarding the compliance and penalties of the contract, the accountability aims for the compliance in terms of workflow execution and trustworthiness of the entities which interact in the achievement of a composed service. Several authors [46] [117] agree to define that a system is accountable if faults can be reliably detected and each fault can be undeniably linked to one or more nodes. In this way, an entity is trustworthy if accountability is guaranteed. For that purpose, techniques such as logging, monitoring and auditing have been proposed.

Ringelstein's work [96] tackles monitor agreed-upon policies in distributed workflows by retrieving information about who, why and how the data is processed. The proposal consists of attaching logs, specified by a RDF-based semantic to all data instances traveling through SOAP messages. That kind of metadata includes data type specification for entity identification (who pro-

cesses the data), action activity (how to process the data) and an ontology to define the purpose of the processing (why process the data). In [117] strong accountability is implemented in the design of the TSOA (Trustworthy Service Oriented Architecture) system, which is able to identify misbehavior, the entity responsible of the fault and evidence of the guilty party. The proposed architecture is based on two main entities: Accountability Service Domain (ASD) and Business Service Domain (BSD), the latter representing the service interacting in the orchestration process. Logging in this work is done by modifying the original BPEL document by inserting XML invoke messages before and after the data is transmitted to any entity, which makes traceability of the data possible. In this way, the accountability service entity in the ASD is in charge of receiving the log for monitoring and auditing purposes. Unlike the previous approach where data used for accountability is centralized, in [46] a different approach in which every node maintains its own log is proposed. In this work, a log containing the exchanged messages is enriched with trusted timestamp data to ensure the facticity of the information used to detect incorrect executions.

Accountability seen from a holistic point of view, covering legal, socioeconomic, regulatory and technical aspects is presented in [89]. That European project named A4Cloud tackles accountability developing tools that: i) enable cloud service providers to give their users appropriate control and transparency over how their data is used. ii) enable cloud end users to make choices about how cloud service provider may be used, iii) monitor and check compliance with user's expectations, business policies and regulations, iv) develop recommendations and guidelines for how to achieve accountability for the use of data by cloud services. A result of this project proposing a meta-model for accountability is reported in [84]. The authors propose metrics for each attribute of accountability: transparency, verifiability, observability, liability, responsibility, attributability and remediation. Such a metrics are derived using evidence and criteria. In this case, evidence may come from several sources (not only monitoring process) including self-assessment from the cloud provision. On the other hand, *criteria* captures contextual information which constraint the metrics (such a policies and contracts). However, the proposed metamodel leaves this last aspect undefined. In [59] it is proposed anonymity as the backbone of the accountability process. The authors propose an anonymity accountability approach composed of several building blocks linked by a communication infrastructure. Authors state that while accountability is required to make entities responsible of their acts, the implementation of anonymity techniques may lead to security risks. Consequently, anonymity and accountability are not opposed, on the contrary, they are complementary.

In general, similar to existing works, I also consider accountability as a key element to trust partners. However, as a consequence of the parameters that are monitored, current approaches mostly consider the detection and assignation of responsibilities for security violations and data misuses. In addition, justification of behaviors (including violations) and propagation of contract breaches are not considered.

5.3 Semi-Automatic Auditing based on Logs

The main contributions of the previous chapters were concentrated on the formal definition of controllability and the representation of those requirements as part of the terms governing the interaction between clients and providers. It led to extend the expressiveness of the SLA by proposing machine-readable service contracts. Such contracts are similar to those in plain English format, in the sense that they can be used to solve disputes in case of claims since they have legal value. Thus, the usefulness of the models proposed in Chapters 3 and 4 can be expressed in terms of formalizing business requirements regarding the use of assets as contractual terms, having a clear understanding of their semantics thanks to their underlying ontology. However, to explicitly define those business requirements as part of a service provision is not enough to ensure their compliance, and it neither exploits the advantages of proposing a machine-readable representation nor a rule-based form for the contractual terms. Indeed, if models and representations understandable for the machine are proposed, it is with the aim of automating some tasks, notably regarding the contract life-cycle.

From the literature review, I have concluded that current methods to enforce non-functional requirements are based on a runtime monitoring process. A controllability approach introduces some challenges which have not been completely addressed in the literature and remain as open issues, particularly in terms of how to verify the compliance with the expected use of assets, when the controller of the asset cannot observe the behavior of the processor and the asset is not anymore in the domain of the controller. In this case a runtime monitoring on the processor side is not a feasible solution since it would be necessary to decompose the business operation into finer actions traceable at the information system level. On the one hand, the processor is not going to be willing to accept that an external organization traces its internal actions; and on the other hand, it might lead to security breaches. Taking this challenge in mind and on the basis of the existing approaches, I propose the use of logs as the main component of an a posteriori[3] method to aim at the semi-automatic validation with the contract compliance. Concretely, the model of controllability based on contracts proposed in Chapter 3 and the model for the contractual terms based on rules proposed in Chapter 4 are used in a knowledge-driven process to verify the compliance with the contract and to evaluate the service provision.

5.3.1 Methodology for the Semi-automatic Auditing

Throughout this dissertation, I have argued the relevance of filling the gap between the technical and the organizational view of a service provision, where the information systems (web services) should be considered as a tool giving support to the organizational needs. In the same way as security policies have been implemented to guarantee some non-functional properties, I have set out the implementation of policies but at a higher-level, i.e. policies representing organizational needs about the use of assets. Thus, from the controllability perspective, a service provision is not only considered as the invocation of some

[3] An a posteriori, as opposed to runtime approaches, aims at the verification of the rule after the provision of the service or after the asset was used, instead of before or during the use.

web service operation or the exchange of messages, but also as the execution of a business process respecting those policies.

Due to the fact that one important limitation commonly associated to the definition of business rules is the difficulty to check their compliance, and even more in runtime, I propose the implementation of a log which is used to check compliance with the contract as a whole after the service is provided or at specific milestones. Therefore, instead of aiming at a real-time decision about the compliance with a specific rule, it is aimed at an a posteriori policy enforcement. To do so, I use the metadata proposed in Chapter 4 as part of the definition of a controllability rule. Such metadata will be recorded in a log to verify the compliance with the contract. As described in the literature review, logs have been widely used for auditing, accountability and monitoring processes. In this work, they are seen as a knowledge base used to draw conclusions about the service provision and the actor's behavior.

In a more detailed view (Figure 5.1), the proposed method begins with the publication of a service description along with its corresponding contract template, which is used as a starting point to negotiate (*step 1*). When a client searches a service in a specialized repository, he/she also sends its requirements regarding the use of assets to starting a negotiation process (*step 2*). If the negotiation succeeds, a WSDL file and an instance of the agreed and signed contract are sent back to the client (*step 3*). Later, when the client sends a request message, he/she attaches some metadata corresponding to controllability rules extracted from the agreed contractual policy (*step 4*). Since both parties have the same contract, each one knows how he should behave and the evidences required to prove compliance with each rule, so when the provider replies to the message, he attaches the expected evidences certifying his/her behavior (*step 5*). Note that the contract terms covers a wide range of assets which not all necessarily apply to the particular request, so it is assumed that the sent rules correspond to the assets involved in the execution of the requested business operation.

The collected evidences and some additional knowledge extracted from the contract are used to create the log. At the end of the service provision, at the contract termination or at some specific milestones, the recorded log can be used with several purposes. For example, an external auditor can compare the log with the contract to verify the compliance with some obligations (*step 6a*). Likewise, the contractual parties can use the log to determine if the counterpart has respected the agreement, but it also gives information to create useful metrics to evaluate the service provision and to give feedback to the contract, for instance in terms of the selection of the provider or to adapt guarantees and expectations (*step 6b*).

Although communication patterns implemented in the service-oriented architectures such as unsolicited pattern, subscription pattern and unsolicited notification do not tie the client to be the initiator of the communication, I assumed above a synchronous request/reply pattern which represents the most common implementation; however, the explained methodology still applies with other patterns. Similarly, note that the steps between the contract template publication and the return of the signed contract have been purposefully omitted since as it was mentioned, the negotiation process is out of the scope of this dissertation.

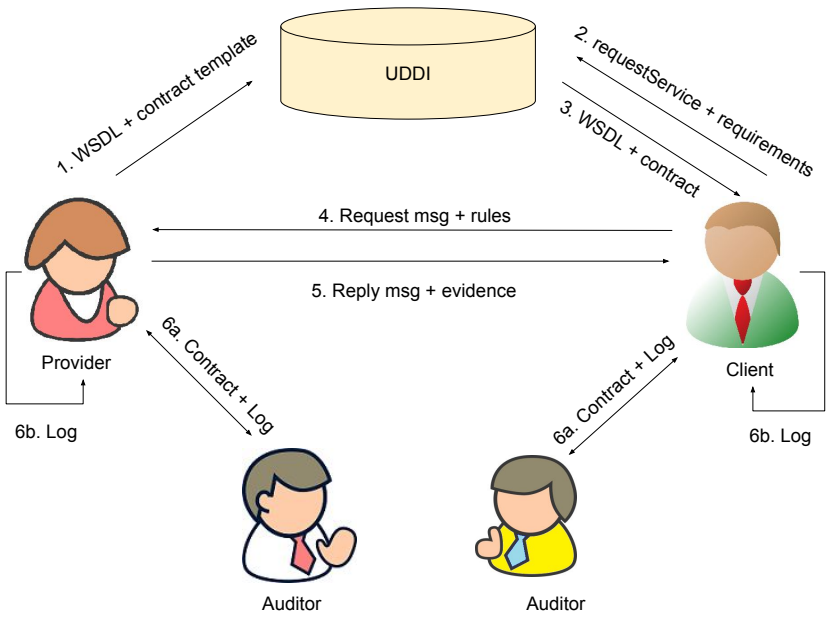


Figure 5.1: Contract Management

5.3.2 Log Creation

While the contract is an agreed document reflecting the terms that regulate the behaviors performed by the contractual parties, the log is a document owned by each organization which reflects what is known about the service provision; that knowledge is useful to make business decisions such as the compliance of the contract, liabilities in case of faults or the trust in the external partner. The log is considered as a knowledge base, but unlike existing works, the log is not proposed under a semantic representation; it means that the restriction of having binary predicates is relaxed since it is not subject to the DL formalism.

As illustrated in Figure 5.1, in the proposed architecture, each partner holds his/her own log where assertions about one's own behavior and the behavior of the external partner are recorded. Although the greatest benefit of the log may be to evaluate the external partner, to record one's own behavior proves worth to automate the auditing process. In this section I will describe the knowledge recorded in a single log and the process leading to each log entry. The following descriptions are valid for both the client's and the provider's log.

In order to draw meaningful conclusions about the service provision and the behavior of the external partner, it is useful to know what happened with each contractual commitment. So, to determine the rule state the following reasoning is made[4]:

- After the contract is signed and its operationalization stage begins, each rule is automatically initialized in a state "*agreed*". It means the fact

[4] In the following part, for the sake for understandability, a predicate-based notation for the log content is suggested, where namespaces and full URI are omitted.

hasState(rule$_i$, agreed, dateAgreed) is asserted in the log, where the *dateAgreed* represents the timestamp information of the time when the contract comes into effect. Together with this assertion the rule metadata presented in the previous Chapters is also recorded:

$$f := \texttt{hasWeight}\ (rule_i, <weight>)$$
$$f := \texttt{hasBeneficiary}\ (rule_i, <Actor>)$$
$$f := \texttt{isSetBy}\ (rule_i, <Actor>)$$
$$f := \texttt{hasAbstractEvidence}\ (rule_i, <AbstractEvidence>)$$

- If a rule is requested to be enforced, a new log entry is created which asserts the rule in the knowledge base together with its state, so the knowledge *hasState(rule$_i$, activated, dateAct)* is asserted as a fact, where the *dateAct* is the timestamp data representing the time in which the rule has been activated (for the controller it is the time in which the message has been sent). Note that even if retraction of facts is allowed in the log, It is not interested to modify the asserted facts but in using the log as a historical archive. Therefore, a new fact with a different state and timestamp data is asserted, where the first one represents the time t_0 when the rule was agreed, and the second one, the time t_1 when the controller asked the processor to enforce the rule.
  Up to this point, the knowledge asserted in the log is composed of the activated rules, their metadata (taken from the semantic contract) and their state:

$$f := \texttt{hasState}\ (rule_i, agreed, dateAgreed)$$
$$f := \texttt{hasState}\ (rule_k, agreed, dateAgreed)$$
$$f := \texttt{hasWeight}\ (rule_k, <weight>)$$
$$f := \texttt{hasBeneficiary}\ (rule_k, <Actor>)$$
$$f := \texttt{isSetBy}\ (rule_k, <ControllabilityActor>)$$
$$f := \texttt{hasAbstractEvidence}\ (rule_k, <AbstractEvidence>)$$
$$f := \texttt{hasState}\ (rule_k, activated, dateAct)$$

  Note that the fact asserting the activation of a rule is recorded in the controller's log at the time of sending the request, while in the processor's side when the request is received. However, in both logs it is asserted that both parties know that the rule has been activated.

Unlike the previous facts in which the request message is used to assert facts about the rule, the reply message is used to assert knowledge about the evidences. Hence, the message is analyzed leading to assert whether some evidence was sent, whether it was required, whether the value of its properties were all satisfied, and whether its properties are enough to prove compliance with the rule or some manual verification is required. Consequently, the following facts are asserted in the controller's log from the automatic analysis of the reply message and the contract. Those facts are also recorded in the processor's log before sending the message.

$$f := \texttt{hasConcreteEvidence}\ (rule_k, <boolean>, dateEv)$$
$$f := \texttt{isEvidenceRequired}\ (rule_k, <boolean>, dateEv)$$

$f :=$ `isEvidenceComplete` ($rule_k$, <*boolean*>, *dateEv*)
$f :=$ `isEvidencePropertySatisfied` ($rule_k$, <*boolean*>, *dateEv*)

Taking in mind that the proposed model considers that an evidence can demonstrate the compliance with several commitments and that rules can be activated several times during the validity of the contract, the *dateEv* element allows to group the metadata of the evidence to a single event represented by its timestamp data.

Since by definition a knowledge base is composed of two elements: facts which assert knowledge, and rules which infer from facts, the following rules are created to automatically determine the state of each rule from the facts.

R1:= If the evidence is sent, its properties are enough to prove compliance with the rule (isEvidenceComplete=true) and all the expected properties of the evidence are satisfied, the knowledge *hasState(*$rule_i$*, accomplished, date)* is asserted. This assertion means that there is enough knowledge to conclude that the rule has been respected. This case means that due to the fact that it is not possible to supervise the external organization, an evidence is required, where all its properties can be parameterized and automatically verified by the information system. Therefore, to verify the evidence is equivalent to verify the rule itself, and since the evidence exists, the compliance with the rule is concluded. An example of this case is to use the value of the IP address as evidence to verify the location of the processor.
By using the symbol `?` to define a variable, and separating clauses per line, this rule is represented as follows:

hasConcreteEvidence(?rule, yes, ?dateEv)
isEvidenceComplete(?rule, yes, ?dateEv)
isEvidencePropertySatisfied(?rule, yes, ?dateEv)
→ hasState(?rule, accomplished, ?dateEv)

R2:= If the evidence is sent, its properties are enough to prove compliance with the rule (isEvidenceComplete=true) and some of the expected properties of the evidence are not satisfied, then the fact *hasState(*$rule_i$*, partially-Accomplished, date)* is asserted. In this case a human intervention is required to determine the impact of such a deviation for the assessment of the contract. An example of this case, is when an e-mail confirming the finalization of some activity is sent but at a later date that expected. In the log, this knowledge is asserted to indicate that the compliance with the rule was incomplete and may need the decision of the expert to determine whether the rules can be considered as accomplished or not.

hasConcreteEvidence(?rule, yes, ?dateEv)
isEvidenceComplete(?rule, yes, ?dateEv)
isEvidencePropertySatisfied(?rule, no, ?dateEv)
→ hasState(?rule, partiallyAccomplished, ?dateEv)

R3:= If the evidence is sent but its properties are not enough to prove compliance with the rule (isEvidenceComplete=no), the knowledge *hasState(rule$_i$, unknown, date)* is asserted in the log. This case is mainly due to the gap between the semantics of the rule and the semantic description of the evidence. In this case, the intervention of the expert is required to manually verify the sent evidence and determine if it is enough to prove the compliance with the rule. To illustrate this case, let us consider an example coming from the French industry. Recently, some companies in France selling products on-line offer their clients to sent a picture of the product before it is delivered to the external shipper. Let us suppose that the picture is requested to prove compliance with some physical attribute of the commodity, for instance, its color. Even if the picture is sent, the property *isEvidenceComplete=no* indicates that it is not its metadata (size, format, date) but the content of the picture which determines whether the rule is respected, requiring the intervention of a human expert the certify the validity of the evidence.

hasConcreteEvidence(?rule, yes, ?dateEv)
isEvidenceComplete(?rule, no, ?dateEv)
→ hasState(?rule, unknown, ?dateEv)

R4:= If the evidence is not sent but required to demonstrate the compliance with the rule, the knowledge *hasState(rule$_i$, NonAccomplished, date)* is asserted. Note that the term non-accomplished is privileged to the term violated since it covers the situation in which the processor behaves according to the rule but he/she does not comply with his/her commitment of sending evidences that confirm it.

hasConcreteEvidence(?rule, no, ?dateEv)
isEvidenceRequired(?rule, yes, ?dateEv)
→ hasState(?rule, NonAccomplished, ?dateEv)

R5:= If the evidence is not sent but not required to demonstrate the compliance with the rule (which can be mainly the case of assignation rules), the fact *hasState(rule$_i$, trusted, date)* is asserted. This knowledge implies that the controller sends some rules to the processor, but he does not need any evidence proving that he complies with those commitments, so he trusts that the processor behaves correctly.

hasConcreteEvidence(?rule, no, ?dateEv)
isEvidenceRequired(?rule, no, ?dateEv)
→ hasState(?rule, trusted, ?dateEv)

Table 5.1 summarizes the state of the rule according to three specific metadata associated to the evidence, namely: whether the evidence was required to

Verified property	Evidence	No evidence
isEvidenceRequired=Y, isEvidenceComplete=Y, isEvidencePropertySatisfied=Y	Accomplished	Non-accomplished
isEvidenceRequired=Y, isEvidenceComplete=N, isEvidencePropertySatisfied=Y	Unknown	Non-accomplished
isEvidenceRequired=Y, isEvidenceComplete=Y, isEvidencePropertySatisfied=N	Partially Accomplished	Non-accomplished
isEvidenceRequired=N, isEvidenceComplete=Y, isEvidencePropertySatisfied=N	Partially Accomplished	Trusted
isEvidenceRequired=N, isEvidenceComplete=N, isEvidencePropertySatisfied=Y	Unknown	Trusted
isEvidenceRequired=N, isEvidenceComplete=Y, isEvidencePropertySatisfied=Y	Accomplished	Trusted
isEvidenceRequired=Y, isEvidenceComplete=N, isEvidencePropertySatisfied=N	Unknown	Non-accomplished
isEvidenceRequired=N, isEvidenceComplete=N, isEvidencePropertySatisfied=N	Unknown	Trusted

Table 5.1: State of the Contractual Rules

be sent (*isEvidenceRequired*), whether its properties need some human verification (*isEvidenceComplete*) and whether all its properties correspond to the expected value (*isEvidencePropertySatisfied*). Note that even if all the cases are represented in that Table, some of them are trivial. For example, a value *isEvidenceComplete=no* overrides the value of *isEvidencePropertySatisfied* because if a manual verification of the rule is required, it is expected that some values of the properties are undefined. Similarly, if the evidence is not sent, the only property considered is *isEvidenceRequired*. Finally, in the case that the evidence is sent but not required, its properties are equally analyzed with the aim of collecting knowledge to determine whether the rule is accomplished or not. For example, if an optional evidence is sent and all its properties are satisfied, it is possible to assert that thanks to the evidence it can be concluded that the rule was accomplished.

In order to verify the compliance with the overall contract, a two-step process is proposed. The first one corresponds to a primitive approach where a fully automatic evaluation of the contract compliance is made based on the weight of the rules and their state. In the second step, the human expert refines the decision making process by manually feeding the knowledge base.

5.3.3 Contract Compliance - Primitive Approach

The compliance with the contract is understood as the organizational process in which the log and the contract are compared (the expected behaviors and the actual behavior) to determine if the contractual policy has been respected. This process is carried out within an organization with the aim of getting feedback of the policy and represents an important stage to evaluate the external partner. It is sometimes referred to as continuous monitoring to denote an organizational procedure wherein the compliance with the internal policies is evaluated.

To evaluate the overall compliance with the contract, it should be considered that a contractual relation can comprise both a long-term and a short-term service provision. Particularly, in a controllability approach considering that the party can adapt his behavior during the contract execution, for instance he fails to send evidences the first time they are requested, but he complies with this requirement in the subsequent times. To tackle this issue, the concept *frame* is introduced, which is associated to the facts asserted in the log. This frame represents the temporal validity for the assessment of the contract compliance. Concretely, the frame is represented as a time interval delimiting the period of evaluation. The definition of the frame is highly important for the contractual parties since it allows to "forget" faulty behaviors performed in the past.

The process of the contract compliance verification begins by determining the rules which are considered for the evaluation. To do so, the timestamp data is used to select only the rules which were in the active state during the specified frame, it means rules which were asked to be enforced during the period of evaluation.

Previously, it was argued that a rule is defined during its life-cycle by a state, which is recorded in the log. To verify the compliance with the overall contract in the primitive approach, the weight of each rule is used. Therefore, the contract state is determined by the significance of the rules belonging to each state at the time of the auditing process. Thus, according to the scale of intensity presented in Figure 4.2, having three obligation rules non-accomplished is more significant than having five recommendation rules accomplished since it is expected (but not mandatory) that obligations have a more important weight than recommendations.

Considering the metrics presented in Table 5.2, the Eq. 5.1 which define the summation of weights of the n rules which are considered to carry out the auditing process, and the Eq. 5.2, which defines the percentage of rules in each state k (k=activated, unknown, non-accomplished, accomplished and trusted), an initial value for the state of the contract can be determined as proposed in the Pseudo-algorithm 1.

$$W_T = \sum_{i=1}^{n} W_{r_i} \tag{5.1}$$

$$P_k = \sum T_k / (T_{act} + T_{accom} + T_{part} + T_{non_accom} + T_{unknown} + T_{trust}) \tag{5.2}$$

$$W_k = S_k / W_T \tag{5.3}$$

Variable	Name	Description
S_{cr}	Total number of contractual rules	The count of all rules established in the policy.
T_{act}	Total number of activated rules	The count of rules with final state activated.
T_{accom}	Total number of accomplished rules	The count of rules with final state accomplished.
T_{part}	Total number of rules partially accomplished	The count of rules with final state partially accomplished.
T_{non_accom}	Total number of rules non accomplished	The count of rules with final state non accomplished.
$T_{unknown}$	Total number of rules of state unknown	The count of rules with final state unknown. Rules for which there is not enough information to automatically conclude a state.
T_{trust}	Total number of rules of state trust	The count of rules with final state trust.
S_{act}	Weight of activated rules	The summation of weights of all rules with final state activated.
S_{accom}	Weight of accomplished rules	The summation of weights of all rules with final state accomplished.
S_{part}	Weight of partially accomplished rules.	The summation of weights of all rules with final state partially accomplished.
S_{non_accom}	Weight of non-accomplished rules.	The summation of weights of all rules with final state non-accomplished.
$S_{unknown}$	Weight of rules with state unknown.	The summation of weights of all rules with final state unknown.
S_{trust}	Weight of trusted rules.	The summation of weights of all rules with final state trust.

Table 5.2: Contract Metrics

Pseudo-algorithm 1 Contract State - Primitive Approach

Calculate W_T according to Eq. 5.1
Calculate S_{accom}, S_{part}, S_{non_accom}, $S_{unknown}$, S_{trust}, S_{act} according to Table 5.2
Calculate the weight of each state W_{accom}, W_{part}, W_{non_accom}, $W_{unknown}$, W_{trust}, W_{act} according to Eq. 5.3
Calculate P_{accom}, P_{part}, P_{non_accom}, $P_{unknown}$, P_{trust}, P_{act} according to Eq. 5.2
If There is at least one obligation non accomplished or at least one prohibition non-accomplished **then**
    Display $Contract_State$ = `Breached`
    Display W_{accom}, W_{part}, W_{non_accom}, $W_{unknown}$, W_{trust}, W_{act}
    Display P_{accom}, P_{part}, P_{non_accom}, $P_{unknown}$, P_{trust}, P_{act}
If W_{accom} has the heaviest weight **then**
    Display $Contract_State$ = `Respected`
    Display W_{accom}, W_{part}, W_{non_accom}, $W_{unknown}$, W_{trust}, W_{act}
    Display P_{accom}, P_{part}, P_{non_accom}, $P_{unknown}$, P_{trust}, P_{act}
If W_{non_accom} has the heaviest weight **then**
    Display $Contract_State$ = `Breached`
    Display W_{accom}, W_{part}, W_{non_accom}, $W_{unknown}$, W_{trust}, W_{act}
    Display P_{accom}, P_{part}, P_{non_accom}, $P_{unknown}$, P_{trust}, P_{act}
If The heaviest weight is $W_{unknown}$, W_{trust} or W_{part} **then**
    Display $Contract_State$ = `Undetermined`
    Display W_{accom}, W_{part}, W_{non_accom}, $W_{unknown}$, W_{trust}, W_{act}
    Display P_{accom}, P_{part}, P_{non_accom}, $P_{unknown}$, P_{trust}, P_{act}

The proposed primitive process concludes the compliance with the contract by determining the heaviest state, as the relation between the sum of the weights of the individual rules in the state k and the total weight of the rules considered in the auditing process. Hence, the most important rules contribute to increment the weight of the state. Note, that if the most representative set corresponds to activated rules, the final state of the contract will be undetermined, since it will mean that at the moment of the evaluation, some rules have been requested but no reply message has been received. If the most representative set corresponds to accomplished rules, it can be inferred that there is enough knowledge coming from the existing evidence to conclude that the contract has been respected.

Similarly, it is pointed out that the semantics of the modal leads to conclude that do not comply with at least one obligation or to perform a forbidden behavior result in a contract breach, even if a remedy was successfully applied. Consequently, it can be perfectly possible that the system concludes that the contract was breached even if the heaviest state corresponds to the accomplished rules. For instance, if four of the most important rules are accomplished but only one obligation is not. In such a case, the evaluation of the faulty party should also consider that he acted correctly regarding the behaviors that were the most important to organization's eyes. As a consequence, in the proposed approach, breaching the contract does not necessarily mean changing the faulty partner since his overall behavior need to be considered.

Basically, at the end of the primitive process an initial contract state is determined and proposed to the expert together with the justification of such a conclusion based on the metrics calculated in the pseudo-algorithm 1. The result of this step can still be used to analyze the rules which have not been not accomplished and to give feedback to the organization, for instance, by proposing the modification of their weight. However, to give a more accurate conclusion, a disambiguation of the trusted, partially accomplished, and unknown rules are given, as well as the confirmation of the non-accomplished rules. This process performed in the refinement step.

5.3.4 Contract Compliance - Refined Approach

A semi-automatic method is by definition a procedure which needs a direct human interaction. In this work, the human intervention allows to manually feed the log with new knowledge about the compliance with a policy. The assertion of such new facts is the result of the human expertise to integrate information which could not be automatically processed by the information system; in particular, the validity of the unknown evidence properties, and the impact of a property value deviation. Indeed, the contract compliance based on the primitive approach was guided by the evidences, but some commitments may be complied with although no concrete proofs exists to demonstrate them. Taking in mind that according to the proposed approach, the lack of required evidences constitutes itself a fault due to the fact that they are part of the definition of the contractual term, some mechanism should exist to fill the gap between the lack of evidences and the actual accomplishment of the commitment. It is operated by using the knowledge of the expert to refine the automated auditing process.

In order to clarify the possible rule state transitions, the rule life-cycle is depicted in Figure 5.2 as a finite state machine.

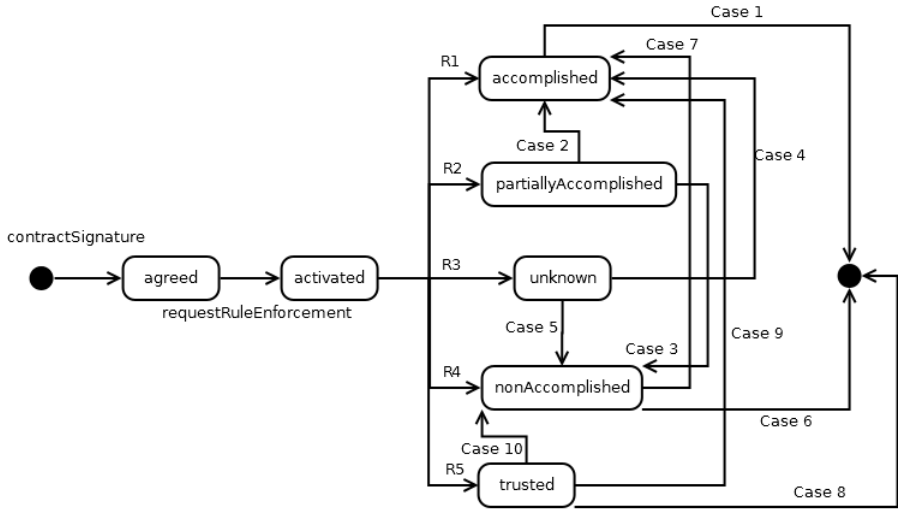


Figure 5.2: Life-cycle of a contractual rule

According to the knowledge expressed above in $R1$ to $R5$, after a rule is activated it transits to a state accomplished, non accomplished, trusted, partially accomplished or unknown, where only one state is possible at a given moment. The states trusted, accomplished and non accomplished are considered as final states because they indicate a normal termination of the rule life-cycle. It means, they allow to determine, based on the evidence, whether the contractual term represented by the rule was respected, or if there is no knowledge, even empirical, which proves its compliance. It is in fact one of the goals of the machine-readable representation. The other states are not considered as final ones since although they give useful information, they do not allow to reach a definitive conclusion about the compliance with the contract. Similarly, it is seen that states can only move forward and the human expertise should be used to overcome the deadlock to which the automated process reaches due to the lack of knowledge to determine the accomplishment with the rule. It should be highlighted that, as depicted in step 6b of Figure 5.1, this particular auditing is performed by the organization itself, which means that all the needed information to determine the compliance with one's own behavior is available, while the knowledge about the behavior of the external partner is inferred from evidences. Consequently, since the objective of this second step is to refine the previous conclusion about the contract state by taking out rules from the state unknown and partially accomplished, it will be expected that all the rules be classified as accomplished, non accomplished or trusted as follows:

- *Case 1*: An accomplished rule does not transit to another state since there is enough evidence to prove the compliance with the rule. This case represents the fully automated auditing of the rule, in which the sent evidences and their properties are enough to demonstrate the correct behavior of the contractual party.
- *Case 2*: A partially accomplished rule moves to an accomplished state.

Rules in a state partially accomplished are the result of a deviation in the expected value of an evidence property. If the expert based on his knowledge and the business goals of the organization conclude that such a deviation is tolerated, the rule moves to the state accomplished.

- *Case 3*: A partially accomplished rule moves to a non accomplished state. In such a case the expert determines that the deviation is not tolerated and the rule changes to the state non accomplished. Note that in the model a threshold of tolerance was not modeled. Although it would allow to automate the transition of rules partially accomplished, the definition of a general threshold with a clear semantics is required, which will be able to differentiate and compute different units of measure. This issue was left as a future work.

- *Case 4* - *Case 5*: A rule in state unknown moves to a state accomplished (Case 4) or non accomplished (Case 5). According to the process defined in the primitive approach, rules can only be in the state unknown if some manual verification of the rule is required. Therefore, the human expert determines from his/her knowledge and the existing evidence, if the rule has been respected (which changes the state of the rule to accomplished - Case 4), or not (Case 5). Note that those rules are not allowed to change to a state trusted since some concrete evidence exists to claim that the organization did its best to behave correctly.

- *Case 6* - *Case 7*: A non accomplished rule does not transit to another state (Case 6), it changes to the state accomplished (Case 7). In Cases 1 to 5, no rule changes to a state trusted since a partially accomplished, unknown or accomplished rule supposes the existence of evidence which can be used to resolve disputes and assign liability. However, rules which are initially in a state non accomplished or trusted suggest the lack of concrete evidence to demonstrate its compliance, therefore if an organization wants to verify the compliance with some rules in the absence of proofs, it returns to the initial problem that is to know if the partner acted correctly or not. Nevertheless, this work has several advantages regarding the original problem. First, since this method is supported in an a posteriori evaluation, some rules can be manually verified, for instance, rules regarding some quality attribute of the commodity. In those cases, the expert asserts a new fact in the knowledge base to indicate that the rule has been accomplished. Note that in such a case the log records the traces that the rule changed from a state non accomplished to accomplished, meaning that the contractual term was verified but the partner failed to sent the required evidence. Secondly, the log can be further exploited to assist the decision making of the expert. Concretely, if the evidence shows that a rule was accomplished, it can be inferred that at some point between the moment when the rule was requested and the sent evidence, the antecedent part of the rule was true and then the consequent. It means, it is a fact that at some point between the moment t_i and the moment t_j the subject had some attributes and some conditions were valid for the business activity or the asset. The refined process consists in determining if the conditions which need to be verified to validate non accomplished rules can be inferred from the known facts coming from the accomplished rules, taking in mind that the period

of time in which this knowledge is considered as a fact should cover the interval of non accomplished rules, as depicted in Figure 5.3. Particularly, this interval is established considering the timestamp data for the log entries `hasState`(rule, activated, $date_i$) and `hasConcreteEvidence`(rule, yes, $date_j$).

Thanks to the representation of the contractual policy in the form of Horn-like rules, individuals facts about the subject, the behavior and the interval of validity can be used to determine whether any non accomplished rule is triggered. In such a case, the compliance with the rule is asserted (Case 7).

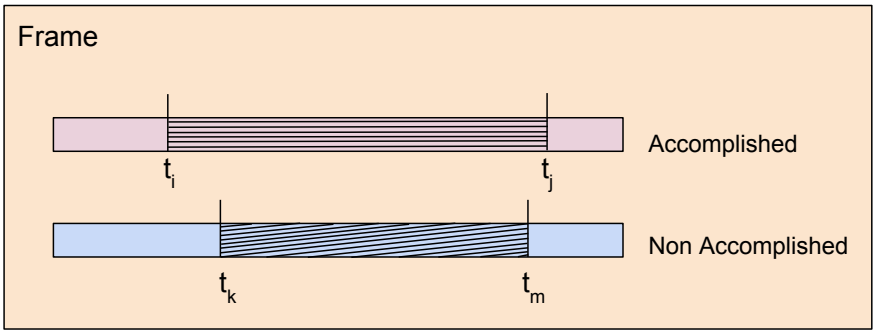


Figure 5.3: Inference about non accomplished rules

In case that the existing knowledge demonstrates that the rule be not respected, the "non accomplished" state will be considered as the final state (Case 6).

- *Case 8*: A trusted rule does not transit to another state. Similar to Case 8, a rule initially in the state trusted does not have direct evidence to demonstrate its compliance. In such a case, if neither the human expertise nor the log can be used to infer new knowledge, the rule remains within the state trusted as its final state.
- *Case 9 - Case 10*: A trusted rule moves to a state accomplished (Case 10) or non accomplished (Case 11). In the primitive process, the only situation leading to consider a rule as trusted is when evidence is not sent and not required, therefore the only way to verify the compliance with the rule is to use the human expertise or the knowledge in the logs as described for the Case 7.

In all previous cases, in addition to the assertion of the new state, if the knowledge used to prove the compliance with the rule differs from the expected evidence and its properties, new facts are asserted in the knowledge base to refine the rule metadata. This allows to get feedback to the policy by proposing alternative evidence and properties to demonstrate compliance with the contractual term represented by the rule. Figure 5.2 summarizes the transitions of the rule state. To improve the readability, the conditions triggering the changes from one state to the next one are represented according to the rules and cases described in Sections 5.3.2 and Section 5.3.4.

Finally, after having semi-automatically determined a state for each activated rule, the process described in the Pseudo-Algorithm 1 is executed again to establish the final state of the contract.

5.3.5 Example - Contract Compliance

In the following, I plan to illustrate the process to determine the contract compliance with an example coming from the petroleum engineering field. This particular application was selected because those industries present complex interactions and are subject to strict controls and regulations.

Let us consider a medium-size oil producer called *Oil & Gas Corporation* (*OGC*) with headquarter located in country A. *OGC* uses *PPMS Enterprise* to carry out the management of its projects (monitoring and planning of the maintenance works). Additionally, due to the fact that the oil plants are located in different countries, *OGC* subcontracts maintenance, repairs and interventions on its plant locally. For the sake of simplicity, let us consider just one plant located in country B, whose operators belonging to the organization *Oil Operator Ltd.* are responsible for physically intervening in the production plant. Moreover, *Oil Operator Ltd.* subcontracts some of its activities when necessary to satisfy the client's demands. Finally, an auditor from the *Certification Office* interacts with *OGC* to evaluate and control the production plant regarding its accomplishment with security norms and existing legislation. Even if one single external organization is described, *OGC* can count several subcontractors (in most of the cases, one per oil plant), the *Oil Operator Ltd.* can operate in several plants, and *PPMS Enterprise* can manage projects of several organizations.

As shown in Figure 5.4, to determine the progress status of the project, subcontracted operators are asked to fill out a form about the progression of individuals tasks and to send it to *OGC*. These data are transferred to *PPMS Enterprise* to calculate global project indicators and modify the initial planning if necessary. This project information is subsequently sent back to *OGC* to support the decision-making process and organizational strategies based on the progression of the job, for instance, to decide if it is necessary to reduce the number of subcontracted operators or to increase the budget allocated to the plant maintenance.

Regarding controllability, some analysis may be done related to: how can *PPMS* prove that the data sent by operators is a factual information?, How to be sure that the form is not filled out at the last minute as a simple formality but that it actually matches the maintenance operations made?, since data is processed in both countries A and B, does *OGC* comply with the existing regulation about the use of data?, How to prove that collected data complies with established policies?, in case that *OGC*'s customers claim for a service breach, who is responsible? the operators? the subcontracted operators?, may the contract breach due to an incorrect project planning?

Note that in such a case, *PPMS* acts as a service provider for *OGC*, but it is *Oil Operator Ltd.* the entity responsible for the collected data that is used by *PPMS* to provide the service (the project management). Therefore, due to the fact that the progress data is part of the *OGC*'s assets, he detains the rights of collection, modification, aggregation and process of such a data. This is a clear example in which a given asset is shared through several organizations, each one using it in a different way and with different purposes.

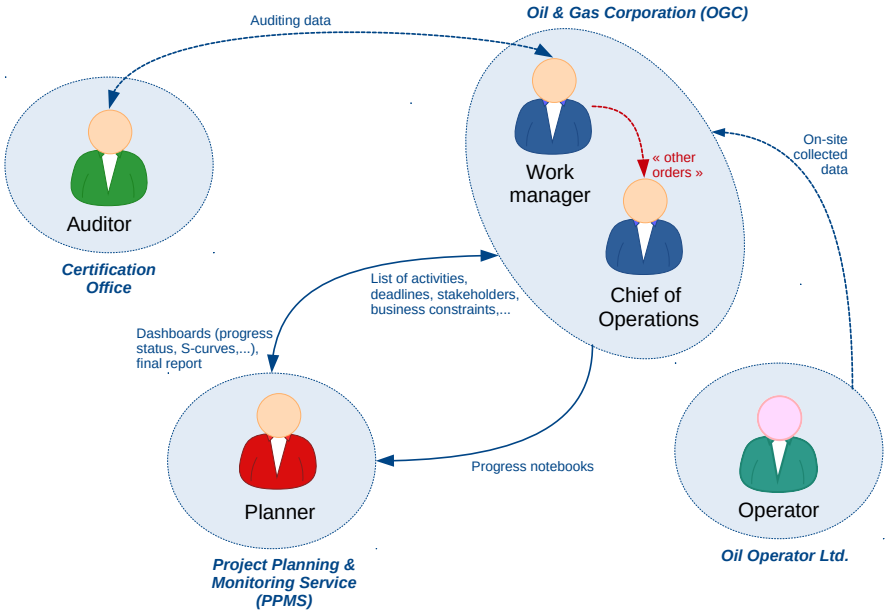


Figure 5.4: Oil & Gas Example

In this example, rules can be of any kind: national/international regulations, business rules or other interaction rules. Just to illustrate the contract compliance process, let us consider that the contract is composed of the following terms belonging to the contract signed between *OGC* and the *Oil Operator Ltd.*:

1. In order to guarantee the accuracy of projects metrics, the progress data collected on-site should to be sent the first day of every month.
2. If the percentage of the project progress is below a certain threshold, *OGC* is compelled to justify the cause of the delay.
3. Operators must attach an electronic signature each time they send project progress data.
4. Operators are restricted to geographical constrains to send project progress data. Thus operators should be in the oil plant in order to avoid that the work is carried out at the last moment.
5. Subcontracting is permitted for delayed activities.
6. The subcontracting personnel is restricted to a maximum number of five new operators ("temporal personnel").
7. In case of subcontracting, "temporal personnel" has restricted access to the plant, having access only to the area associated to their activity.

In Chapter 3 an example of a contract header belonging to this same field was presented, and it was complemented in Chapter 4 with an example of a

contractual rule, which was transformed from the natural language to their XML representation. Since in this Section I have been focused on the contract evaluation, it is assumed that the contractual terms presented above have been successfully transformed into a XML rule syntax. It is a justified assumption because both the machine readable representation of rules and the methodology followed to do such a transformation were previously illustrated. Then, only the log will be presented where full IRIs have been omitted, and the rule R_i is the i_{th} contractual term described above.

It is highlighted that the aforementioned process for the contract evaluation is not tied to any specific technology. Due to the fact that it is presented as a pseudo-algorithm instead of a concrete syntax, it can be implemented in almost any specific language with a greater or lesser degree of simplicity. Although several concrete languages can be used to create knowledge bases, including XML, OWL and SQL, I will illustrate the proposed method by representing the log in two formats, namely the RuleML language, and CLIPS, which allow to illustrate two different approaches to implement the proposed method. The choice of RuleML was motivated by the fact that this representation proposes two syntax, namely, XML, which allows an easy integration with other technologies, and the POSL[5] (Positional-Slotted Language) syntax, which is easier to read, compact and similar to the abstract representation in which this process was presented. In general, RuleML supports ontology knowledge, as well as queries and inferences based on the top-down as well as bottom-up manner. Similarly, several parsers have been developed and integrated to current engines to translate POSL facts and rules into XML and vice versa. On the other hand, CLIPS is a LISP-based language used to develop expert systems. CLIPS supports logic programming and OO features; it overcomes some of the limitations found in a log based on RuleML, in particular counting and accumulation, which are both important computations to calculate contract metrics.

For this example, it was assumed to give the first tests a frame of four months (from September to January), where the facts corresponding to the rule state have been asserted according to the procedures described in the previous Section. The metadata associated to each rule is presented in Listing 5.1. The rules to calculate the state of each contract term after they have been activated as well as the state of the overall contract are also included in the knowledge base.

Listing 5.1: Rule Metadata Oil&Gas Example

```
hasWeight(R1, 0.3).
isSetBy(R1, OGC).
hasAbstractEvidence(R1, sentDate).
hasWeight(R2, 0.7).
isSetBy(R2, OGC).
hasAbstractEvidence(R2, delayJustification).
hasWeight(R3, 0.8).
isSetBy(R3, OGC).
hasAbstractEvidence(R3, digitalSignature).
hasWeight(R4, 0.54).
isSetBy(R4, OGC).
hasAbstractEvidence(R4, Location).
```

[5] http://ruleml.org/submission/ruleml-shortation.html

To make tests, it was supposed that each rule be requested three times during the specified frame, where for rules `R1` and `R3` evidences were successfully sent, and for rule `R2` the evidence was not sent the second month.

I will begin by explaining the process and tests made in RuleML. In order to illustrate useful queries to the knowledge base, two examples are described which are performed using the top-down inference of OO jDREW. First, after the rules (contractual rules and rules inferring the state of the contractual terms) and the set of initial facts are asserted in the log, a query is made to know what happened with rule `R2` during the validity of the contract (`hasState(R2, ?state, ?date)`). So the result to this query represents the life-line of the rule. As shown in Figure 5.5 (different results were superposed in the same Figure but for space limitations not all the results were shown), it is possible to know that such a rule was agreed on September 12th, activated on October 01st, trusted for the request made in October, and activated again on November 01st.

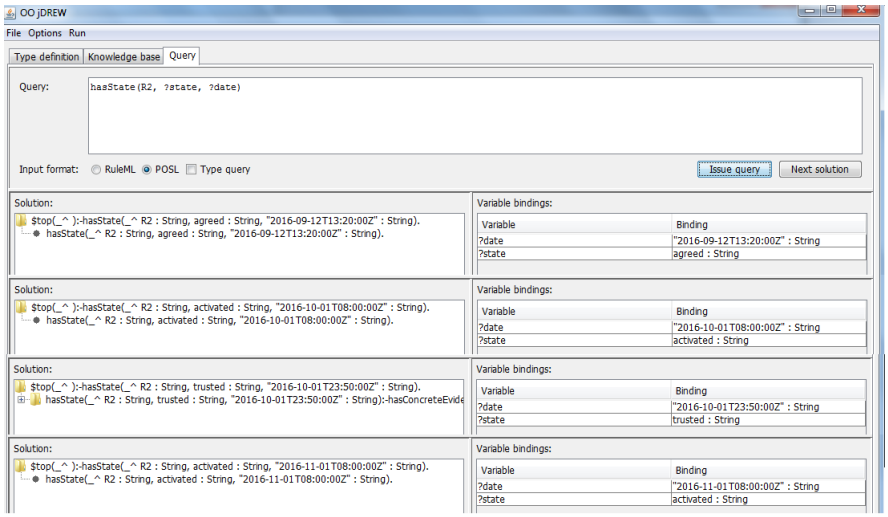


Figure 5.5: Life-line of the Rule R2

In the second example, the knowledge base is queried to list the rules which are in a particular state. Concretely, I want to known which rules are in a state trusted (`hasState(?ruleIdentifier, trusted, ?date)`). As shown in Figure 5.6, the system returns the rule R2, which is consistent with the previous query. Moreover, the system justifies that the rule was considered as trusted since no evidence was sent but at the same time the evidence was not required.

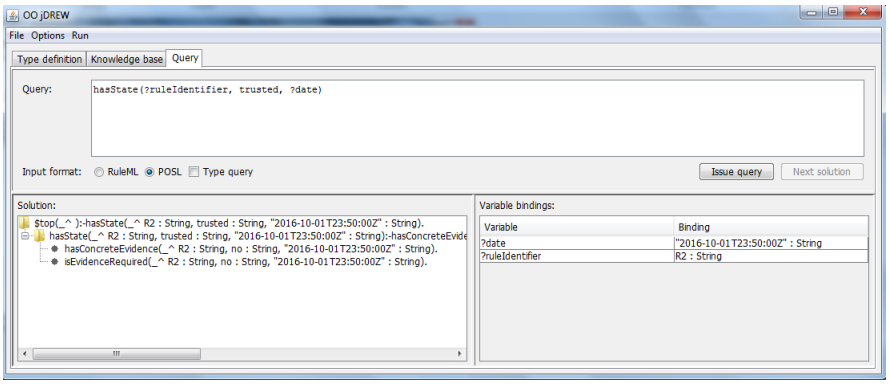


Figure 5.6: Query about Trusted Rules

After operating tests and queries to extract useful knowledge about rules, I have focused on analyzing the contract as a whole. To do so, it is requested to calculate the metrics presented in Table 5.2. Despite the advantages of the chosen representation, some procedures are easier to represent in imperative language than in declarative one (for instance accumulator variables, counter variables and global variables). Although RuleML proposes several built-ins for comparisons and math operations, to the best of my knowledge no built-in exists for counting or accumulator. To overcome this limitation several solutions exist. Firstly, OO jDREW is well documented regarding the creation of built-ins, which can be used to integrate counting operators; this is similar to the way as SWRL integrates SQWRL operators. Secondly, an alternative approach may be to directly use the XML syntax and use languages such as XPath to query the knowledge base. Figure 5.7 shows on the left the knowledge base in POSL syntax, and on the right an example of its automatic transformation into RuleML XML made with OO jDREW. The simplest solution is to separate declarative knowledge from iterative operations, where the log does not contain the full representation of the process but is used as input to make some computations which will be later asserted as facts in the log, such as the percentages and the accumulated weight by rule state. The drawback of this solution is the need to integrate different technologies to the complete implementation of the process. As it will be discussed later, other languages for the construction of knowledge bases, such as CLIPS, allow to directly calculate the sum of weights as well as counters, which allows to easily determine the percentages and the weights presented in Eq. 5.2 and Eq. 5.3. I highlight the fact that this step is the result of the chosen language and that it does not represent a limitation of the process itself.

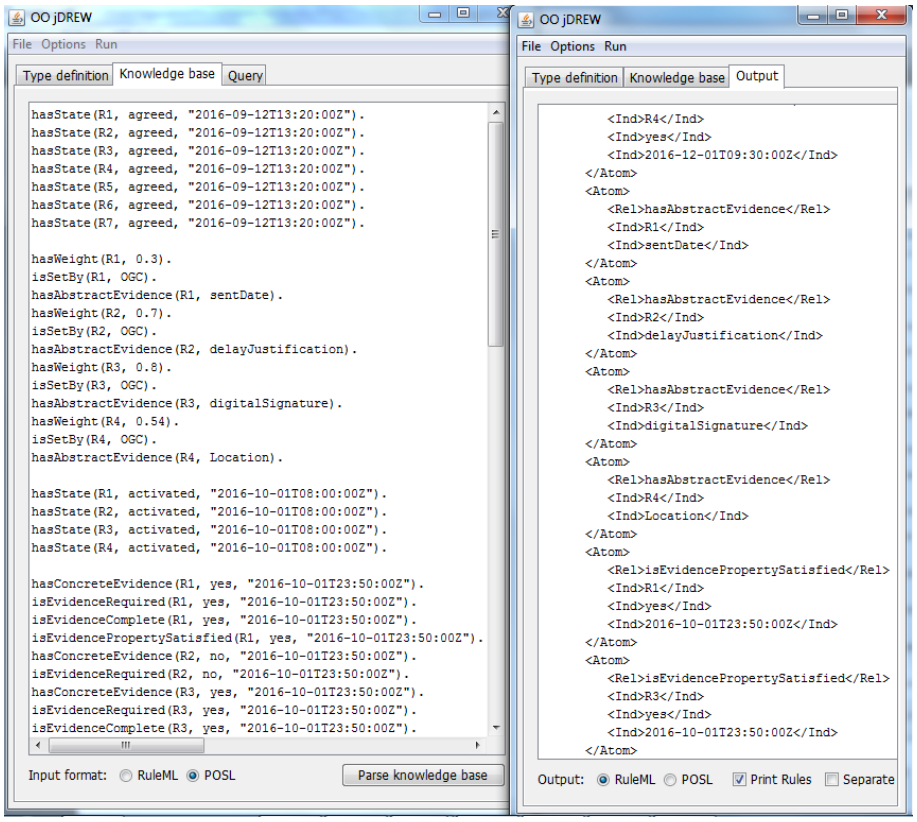


Figure 5.7: Facts in the Log Knowledge Base

Figure 5.8 shows the result of the primitive approach, considering a frame from the 01st September to 01 January. The result of this auditing is that during the established period, the contract was accomplished. On the left, the system explains that this result was reached since the weight of the set of rule accomplished is greater than the weight of the another of states.

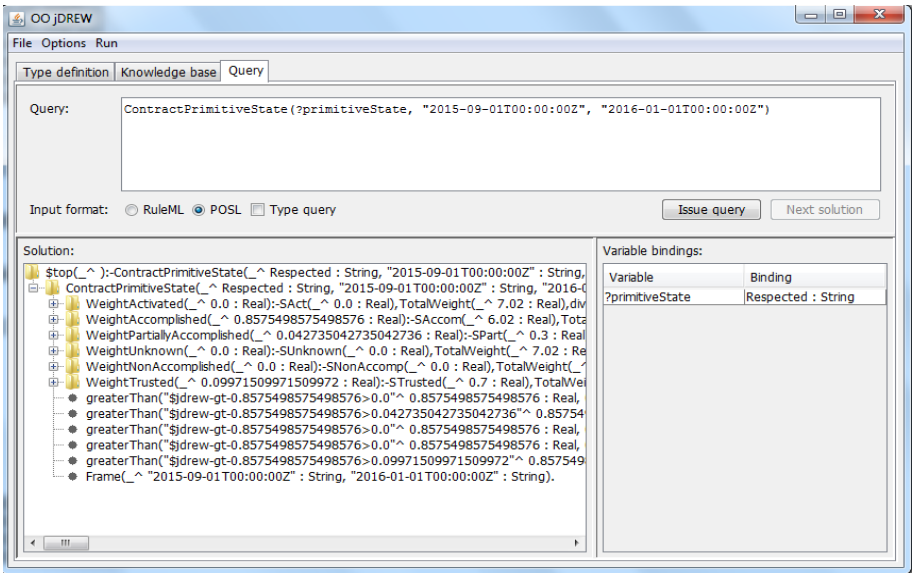


Figure 5.8: Automatic Contract Evaluation in RuleML

By continuing with the experimentation phase, the knowledge about the contract evaluation is represented in CLIPS. This language is highly expressive to support both declarative knowledge (facts and rules), and object-oriented programming. It facilitates the fully implementation of the proposed method in one single knowledge base. Figure 5.9 shows the resulted list of facts. Note that the prefix notation ("Polish notation") of CLIPS is similar to the POSL syntax of RuleML. Indeed, the ordered representation of facts is composed of templates defined from slots. It is shown that for the seven terms composing the contract, there are 42 facts asserted in the knowledge base, and after the system is run, new facts are automatically inferred regarding the state of rules. When an automatic auditing is requested to the knowledge base, the system calculates weights and percentages of the rules in the defined frame and asserts this data in the log. Finally, it presents the conclusion together with the contract metrics. Figure 5.10 shows that for the frame between 2015-01-01 and the 2016-01-01, the system concludes that the contract was respected, where the 83% of rules were accomplished corresponding to the 0.85 of the total weight of evaluated rules. I can also notice that the fact 55 (f-55) represents the contract metrics of that specified frame.

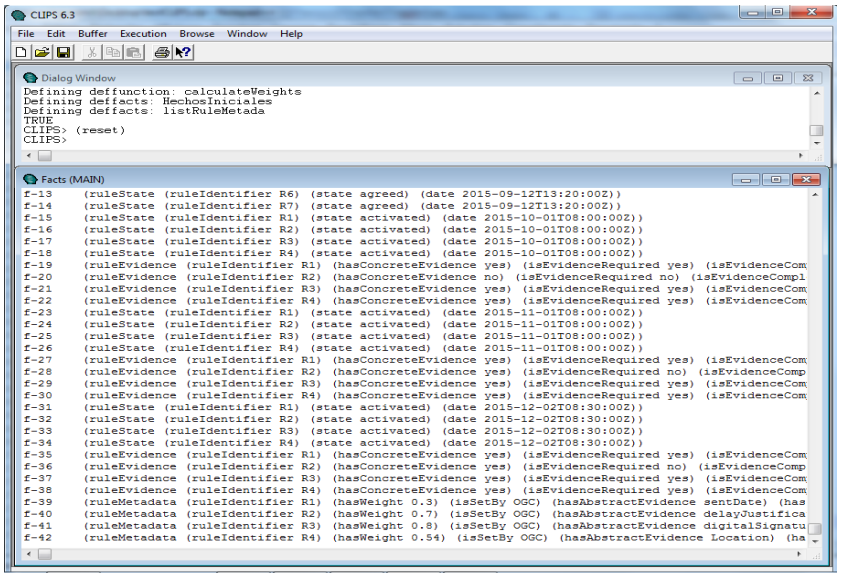


Figure 5.9: Knowledge Base in CLIPS

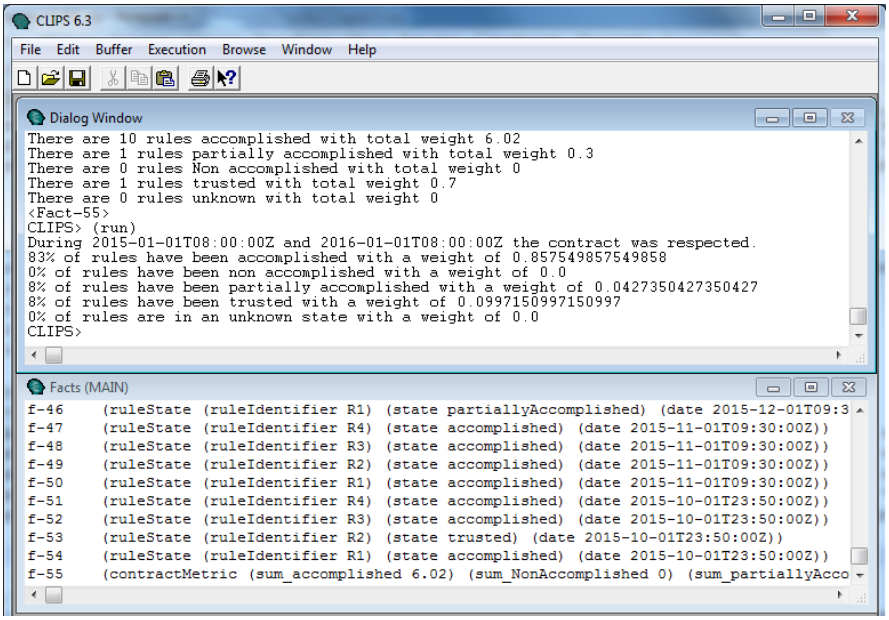


Figure 5.10: Automatic Contract Evaluation in CLIPS

Some other tests were performed considering different situations such as changes in the evidences both in the frame of evaluation and in weights. Figure 5.11 and Figure 5.12 present the results of two of those experiments. In the first one, the knowledge about the compliance of the rule R3 is changed by indicating that when such a rule was requested for the second time, no evidence was sent proving its compliance. As shown in Figure 5.11 the result of the automatic auditing shows that the contract was *breached* and a justification associated to the non accomplishment of the rule R3 is presented. Note that this conclusion is drawn even when 75% of the rules have been accomplished. Considering this non accomplished rule, the Frame of evaluation was changed by considering only the last month, from November to December. For this case the contract is considered as *respected* since the non accomplished rule (R3) took place on November 11th, and the auditing began on November 30st. The misbehavior of the partner was "forgotten". Figure 5.12 illustrates the result of both evaluations.

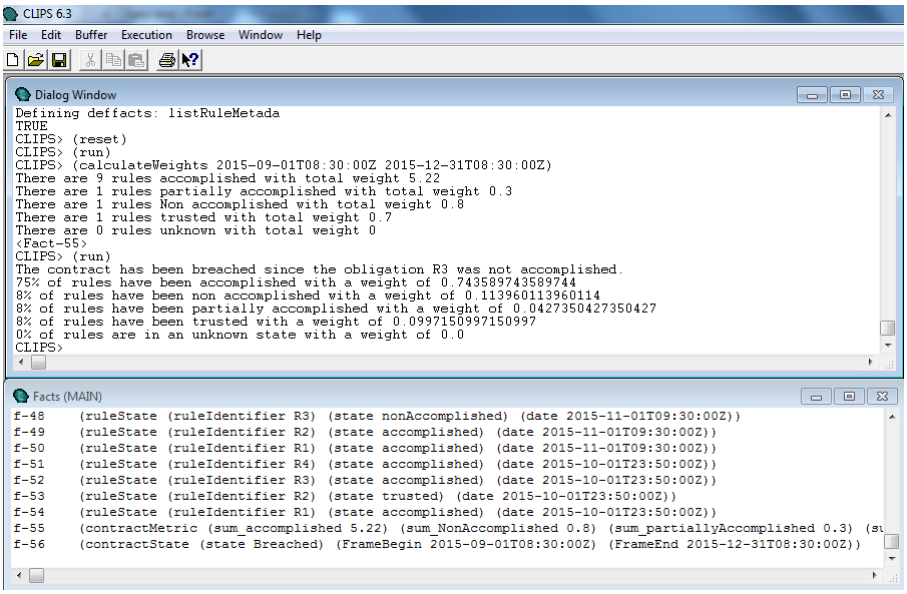


Figure 5.11: Automatic Contract Evaluation - Breached Contract Case

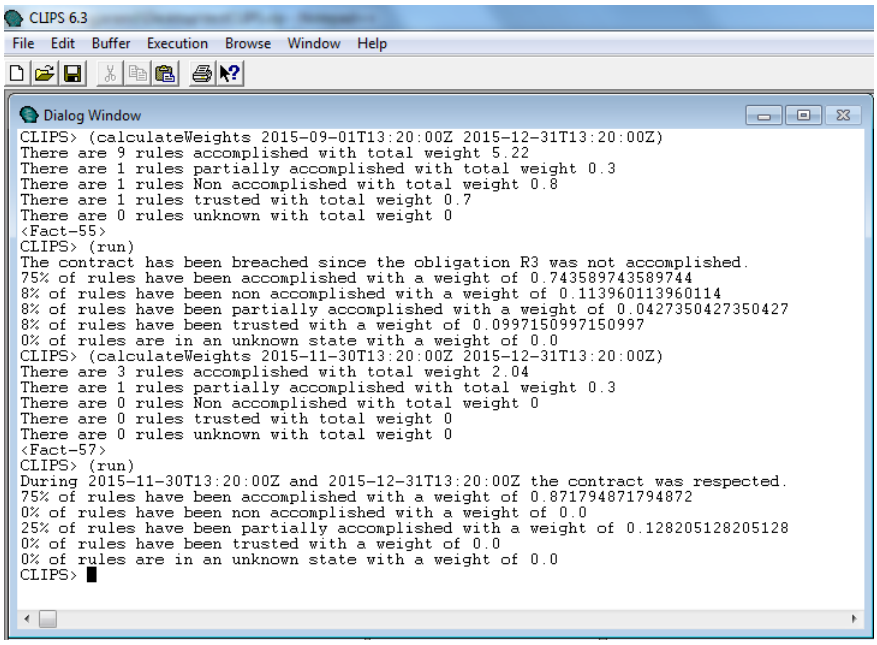


Figure 5.12: Automatic Contract Evaluation - Frame Change Case

5.4 Knowledge Base Analysis

In this dissertation, I have argued the usefulness of an a posteriori evaluation of the contractual relation from the business point of view. Indeed, it supports organizational processes such as the automatic verification with the contract compliance, the assessment of the external partner, audits verifying the compliance with existing regulations (by the use of evidences), and the assessment of internal policies. Moreover, a heuristic evaluation is in phase with the general specification of a service contract assessment [114], which considers that some activity or commitment can be performed late, it is modeled in this approach by *partially accomplished* rules. Since it requires a comprehensive evaluation of the contractual party behavior, this situation cannot be detected by a runtime monitoring.

In the previous sections, I have described the content of the log and the following process to determine a state for each rule. The knowledge represented in the log was used as input to reasoning about the contract compliance in what can be called an "intra-contract" analysis since one single contract (corresponding to a single service) is involved. In this approach, other analyses can be done taking as input both the log (which contain the available knowledge about what happened during the service provision) and the contract (which contains the policy and the semantics of the contract's elements).

First, as it was previously described, the first usefulness of keeping trace of what happened during the service provision is to use the asserted knowledge to try to infer whether rules in a state unknown or trusted (rules for which no direct evidence is available) were accomplished. Indeed, as depicted in Eq. 5.4, from a bottom-up reasoning it can be demonstrated that if u is asserted in the

log, there is a path (a limited set of conditions) leading to such an assertion.

$$u \leftarrow (p_1 \wedge p_2 \wedge \cdots \wedge p_n) \tag{5.4}$$

Due to the fact that rules cannot directly be evaluated (since they represent abstract activities), the consequent will be asserted in the log iff the rule is evaluated as accomplished, and by definition the consequent can only be true if the antecedent is true. It leads to infer that at the time between the activation of the rule and the sending time of the evidence, the actor and the asset were in a particular state (by exposing some attributes). Consequently, if some other rules need to verify the same subject (actor and his attributes), behavior (activity/asset and its attributes) or context, the available knowledge can be used to support their compliance. It demonstrates the usefulness of a rule-based representation. Note that this step is part of the refined process, which means that inferences are expected to be validated by human expert. Moreover, a complete validation of the antecedent is required, where the subject, behavior and context can be demonstrated by concrete evidence.

As a result, the first feedback to suggest to the organization is the new evidence which has been used to guarantee the accomplishment of the rule:

$$NewEvidence = Ev_{sub} + Ev_{beh} + Ev_{ctx} \tag{5.5}$$

In this same analysis, a second feedback is associated to the knowledge that the expert used to assert that a rule which was partially accomplished, moves to a state non-accomplished or accomplished. Indeed, if it is known that some attributes of the evidence are important enough to determine the state of the rule, the value of the property `isRequired` should be set to `true`. In general, in this second feedback, the knowledge leading to justify why a rule changes of state in the refined approach was presented. Let us recall that when the expert makes an assertion, the knowledge used as evidence is asked.

Secondly, the result of the (semi-)automatic process of the contract compliance can be used to determine the assets affected by the non-compliance with a rule. From the output of the process presented in Section 5.3.3 and Section 5.3.4, the list of non-accomplished rules is known. Therefore, the following reasoning can be made to determine the organizational resources compromised by the misbehavior of partners:

Rule(?r)
hasState(?r, NonAccomplished)
hasBehavior(?r, ?bh)
hasConcreteBehavior(?bh, ?c)
employs(?c, ?a)
→ CompromisedAsset(?a)
compromisedBy(?a, ?r)

The above reasoning is valid both when ?c is a business activity (since by definition a business activity `employs` an asset) and when the concrete behavior is an asset which interacts with another resource. Moreover the notion of *second compromised asset* is proposed (Eq. 5.6) to model the cascading effect of affected assets. An organizational resource which interacts with an affected asset can also suffer from the consequences of the non-compliance with a rule.

$$\begin{aligned} &\text{CompromisedAsset}(?a) \\ &\text{compromisedBy}(?a, ?r) \\ &\text{employs}(?a, ?ca) \\ \rightarrow &\text{SecondCompromisedAsset}(?ca) \\ &\text{compromisedBy}(?ca, ?r) \end{aligned} \tag{5.6}$$

Thirdly, if the contractual terms represent the expectations (wishes) of the parties about the service provision, they can be used to evaluate the quality of the service. In [47] the quality of any service is proposed to be evaluated by taking five dimensions into account: assurance, tangibles, empathy, responsiveness and reliability. When this latter is defined as the ability of a service provider to deliver the promised service dependability and accurately. As it was previously stated, the contract can be seen as the contractual parties' expectations while the log is the actual provided service. Therefore, since the contractual terms represent a category for the rule, it is suggested to calculate a metric for each of the category presented in the contract to which a rule is associated (see Figure 3.9), i.e. remedy, payment, claim, legislation, cancellation, liability, guarantee and exclusion. Consequently, instead of evaluating the service only in the terms of guarantee terms specifying performance and security properties, the compliance with the complete contract is considered. Previously, it has been established that one of the implications of the inclusion of the business perspectives is to broaden the parameters used to assess the quality of the service. Indeed, for an organization, it is not only important to secure and make the web service effective (acting as the tool mediating the interacting between clients and provider) but also to measure the quality in terms of the satisfaction regarding the use of resources and the expected behavior of partners. Although more sophisticated metrics can be implemented, I consider the average as main metric:

$$\text{MetricCategory}_x = \frac{\sum \text{hasState}(\text{rule}_{ix}, \text{accomplished})}{TotalCategory_x} \tag{5.7}$$

This evaluation of the service goes hand in hand with the partner's evaluation. In this case who is the processor of the asset should be considered (`isUsedBy` properties). Although a complete model of trust or reputation is out of the scope of this thesis, I suggest that the list of compromised assets be used to quantify the way in which the external partner acted during the service provision.

$$\text{Performance}(proc, cat_X) = \frac{\sum(isCompomisedBy(?a, ?r), isUsedBy(?a, ?proc), type(?r, ?X))}{\text{TotalCat}_X} \tag{5.8}$$

Such a metric is contextualized by indicating the total weight of the rules associated to the processor.

$$\begin{aligned} Weight(proc, cat_X) = SumWeight(&compromisedBy(?a, ?r), isUsedBy(?a, ?proc), \\ &hasWeight(?r, ?w), type(?r, cat_X)) \end{aligned} \tag{5.9}$$

Both represent the performance of the processor grouped by category and also the total relevance of the commitments associated to the processor in such a category. It allows to conclude for example if the most important rules in the controller's eyes are successfully complied.

Finally, regarding liabilities, since the modeled contract is restricted to having only two parties, if some contract deviation exists, the responsibility falls to the subject of the rule; However, in this case, other kind of analysis can be made to draw useful organizational inferences. Such a process can be defined as a "cross-contract" evaluation. Unlike the previous analyses based on the knowledge recorded in a single log, several logs can be merged in order to determine if patterns can be found regarding non accomplished rules, for instance determine a profile of user who repeatedly infringe the contract or to determine what is the most frequent infringed rule. Similar to the previous cases, such a metrics can be calculated by querying the log in order to identify the frequency of violated rules belonging to different contracts. Moreover, a threshold can be set to identify if the such a counting is higher than a given value.

To summarize, it should be noted that the method proposed in this Chapter mainly relies on heuristic knowledge drawn from evidences. It highlights the importance of its correct definition and due to the fact that they are explicitly included in the contract, they become part of the negotiation process. Similarly, it is highlighted the role played by the delimitation of the frame, since as it was shown misbehaviors performed by a partner can be forgotten, which in turn modifies the metrics associated to the processor.

5.5 Conclusion

In the service-based paradigms, the expressiveness of the machine-readable SLA allows to represent guarantees about the service provision in terms of security and performance parameters. Thanks to the characteristics of such parameters being observable and measurable, they have favored automated processes such as the dynamic reconfiguration of workflows and the runtime verification of the compliance with the expected service level objectives. Otherwise, the representation of business requirements had been usually focused on monetary terms, leaving aside issues which can result in organizational risks such as the use of assets. In this Chapter, I have not only shown the importance of explicitly representing guarantees about the use of assets, but also that a runtime monitoring is not always the most advisable option to verify the compliance with those controllability rules. Thus, I have suggested the evaluation of the compliance with the policy at the end of the service provision and not during the use of the asset that I have called an a posteriori approach. I have argued that this approach, although not reactive, becomes an important process within organizations since it allows to support the decision-making process considering the overall behavior of the contractual party.

The a posteriori process uses contract information and evidences to give information about what really happened during the service provision and also allows to calculate contract metrics. This verification is highly important for auditing, decision-making and risk management processes. Indeed, collected metadata can be used to check compliance with regulations, to assign responsi-

bilities for contract breaches, calculate indicators such as party's reputation or to identify patterns leading to prevent contract breaches due to misbehaviors. In particular, in this Chapter a log containing knowledge about the behavior of each partner is used to support a semi-automatic process leading to verify the compliance with the contractual terms and to evaluate the performance of the contractual parties.

The analysis of the proposed heuristic process and the definition of the log as a knowledge base lead to interesting perspectives about the use of artificial intelligence techniques to analyze the knowledge captured in the log, in particular machine-learning and data mining techniques. Hence, considering the evaluation of the contract compliance as a classification problem, artificial neural networks can be used to adapt the value of the weights of each rule and to use cross-contract logs as a historical archive to determine future contract breaches.

Part IV

General Conclusion

Chapter 6

Conclusion

The general meaning of the provision of a service refers to the delivering of a tangible or intangible commodity to somebody. The service-oriented paradigms have favored the computerization of that process by making the service accessible to a wide range of users, reducing the needs of technological homogeneity and automating tasks such as the authentication, authorization, provider selection and the verification of some performance parameters.

Supported by the literature review, the importance of explicitly establish the terms governing a service provision has been argued. First, it means to match the client's expectations with the actual provided service. Secondly, it allows to create the foundations of a trusted relationship between the client and the provider, where each party knows what is done during the service provision. Thirdly, it serves as support to evaluate the compliance with some commitments. In particular, organizations have realized the need of clearly specifying in those terms what each partner can, cannot, should or must do during the service provision. Indeed, in the B2B context, an implicit "business dependency" is generated caused by the fact that some assets are shared between clients and providers. The way they are used by the external partner may cause organizational damages such as loss of clients, financial penalties, loss of reputation and lawsuits. This work was particularly focused on shared assets since they move from an organizational domain to other, where the business activities in which the asset is involved define its usage. It is, in fact, a challenging issue emerged in the service-oriented technologies because organizations are interested in preventing damages which may be caused by misbehaviors of the external partners, but there is no way of knowing if the asset has been correctly used since it has left the domain of control and the processes of the external organization are considered as black boxes.

This dissertation addresses the problem of the lack of control in the use of assets shared during a service provision, which has been defined as the problem of *controllability*. Even if it is a valid problem in B2B as well as B2C contexts, it was assumed a B2B scenario since it allows to take in consideration the widest scope in which both clients and providers want to keep control on their respective assets. It has been also shown that current approaches aiming non-functional service guarantees do not satisfy the need of representing business requirements about the expected use of assets.

In the last years, the academic community has taken an increasing interest

in researching about the human behavior as a relevant element of security. Even if partners collaborate during the service provision, they can still have opposed objectives, and besides, people can sometimes have an irrational behavior; so considering the use of assets as a facet of the partner's behavior is not a trivial issue. Throughout this dissertation, I have argued and demonstrated that modelling the behavior of partners by using controllability policies is feasible along with the use of the available knowledge collected during the service provision, to verify the compliance with such a policy. Thus, the expressiveness of current machine-readable service agreements can be extended to aim controllability requirements. Following, the contributions of the dissertation are analyzed in the light of the objectives below. Later, the perspectives of this work are described.

6.1 Objectives and Contributions

- **To choose a machine-readable representation** able to formalize business guarantees about the provision of a service.

Having in mind the general objective of representing controllability as part of the non-functional requirements governing a service provision, in Chapter 3 the review of the literature was focused on current approaches aiming to represent guarantees about the provision of a service in a machine-readable form. The research in this area shown two main approaches, namely: certificates and SLA, which were analyzed to determine whether they can be adapted to support business requirements. Although adaptation is possible, the nature of the changes were such that the principle of the current approaches itself needed to be rethought. Indeed, documents written in plain-English regulating the provision of a service were used as basis to identify the needs in terms of representation, and several findings led us to conclude that an alternative approach was necessary. First, organizations define restrictions on a wider range of assets than only data. Secondly, business requirements regarding the use of assets are expressed in terms of coarse-grained organizational activities instead of fine-grained actions. Finally, since both assets and activities are represented using a business vocabulary, a clear semantic is needed in order to avoid misunderstandings. Therefore, not only the expressiveness of the current approaches need to be extended, but also the mechanism of verification since controllability requirements tackle activities which are not observable or measurable in runtime.

Thus, I have proposed in Chapter 3 the definition of machine-readable semantic contracts to represent business requirements governing the provision of a service, whose modeling is the result of a tagging process made on a set of real-case service contracts written in plain English. It has been shown that the $\mathcal{SROIQ}(\mathcal{D})$ formalism offers the required expressiveness to define the specialized concepts belonging to the domain of knowledge. The advantage of such a formalism is its robust expressiveness and its decidability, which makes it suitable to a machine-readable representation. In particular, the OWL language and its XML concrete syntax were used to represent the semantic contract.

The proposed representation results in a generic model of service contracts. Although some other semantic representations have been proposed in the service oriented field, to the best of my knowledge the contractual relation between clients and providers had not been considered. It becomes therefore the first contribution of this work.

- **To propose a formal definition of controllability** as part of the non-functional business requirement governing the relation between service clients and providers.

Controllability has been defined as the control of the use of assets. In the literature, usage control approaches have already been proposed by the UCON models; however, in those works the usage is defined in terms of the continuity in the access. It means, an ongoing evaluation in which the granted rights can be revoked during the use of objects if some properties are not longer valid. In [68] controllability is considered from a business perspective, and specifically, the use of assets is associated to risk management. Nevertheless, the authors of that work do not propose any formal definition of controllability and it is only presented as a new abstract security property.

The second contribution of this dissertation is a semantic formalization of the control of the use of assets, where the controllability vocabulary comes to enrich the contract model. Such a vocabulary defines the controllability actors as well as an asset taxonomy which is consistent with the ISO/IEC 27005, and the definition of business activities in terms of organizational operations performed in those assets.

- **To propose a model for the controllability requirements** which will be able to represent the expected behavior of an external partner regarding the use of assets.

Since the conclusion of the first objective was to represent business requirements in a service contract, it is clear that the definition of controllability requirements becomes part of the contractual terms binding clients and providers to accomplish some commitments. The literature review presented in Chapter 4 focused on current approaches to represent policies, in particular, those involving several organizations. Three main findings led to propose a specific model to represent controllability policies. First, unlike the proposed approach, most of the current inter-organizational policies assume that the asset remains in the domain of control, where some rights are granted to external partners. Secondly, the objects and actions defined in the reviewed policies are restricted regarding the organizational needs of controllability. Finally, policies are aimed to be enforced in runtime, which cannot be possible to do with controllability rules since the actions of the external partner are unknown, and hence, not monitoreable in order to maintain the inner logic of each organization.

Considering that the way each organization uses the assets defines its behavior, in Chapter 4 I proposed a model for the controllability rules composed of four main components, namely: a subject, which defines an actor and some particular attributes; a behavior, which defines a business activity and its conditions; the context, which defines system and environmental conditions; and finally, the modality, which indicates if the expected behavior corresponds to a permission, a recommendation, an obligation or a prohibition. Particularly, the meaning of the elements described in those components is taken from the semantic model presented in Chapter 3. It aims to overcome possible misunderstanding between the contractual parties due to the use of a business vocabulary. Besides the structural definition of rules, some metadata complement their semantic description to represent the organizational relevance of the defined usage, which is modeled as the *weight* of the rule; similarly, since the rule cannot be directly

verified, a machine-readable *evidence* is defined to demonstrate the compliance with the expected behavior; lastly, the *purpose* of the rule is modeled.

The representation of machine-readable contractual terms based on the proposed model contributes to extend the expressiveness of current SLAs with service guarantees about the expected behavior of actors. Those guarantees are able to express more complex requirements associated to the business needs than only security and performance guarantees.

- **To verify the compliance with the controllability rules** in order to evaluate the performance of the service actors and the quality of the provided service.

In Chapters 3 and 4 the model part of the proposed method was presented. It was composed of two representations, the first one modeling the semantics of the service contracts, including a controllability vocabulary, and the second one modeling controllability rules as part of the contractual terms. In Chapter 5, I argued the need of verifying the compliance with the contractual rules as a relevant stage of a governance approach. Indeed, describing explicitly the needs of controllability neither guarantees their compliance nor leverages the advantages of a machine-readable representation of contracts compared to plain-text representations. The main challenge of the policy verification is that the price to pay to extend the expressiveness of the service agreements is to represent activities and attributes which cannot be monitorable or measurable in runtime, which makes difficult to evaluate their compliance through traditional monitoring techniques. In Chapter 5, a process to verify compliance with the contract is proposed. It uses the metadata, and particularly, the weight of each rule and evidence, to create a log used as a knowledge base to infer the state of each rule, and consequently, to determine if the contract has been breached, respected or if there is not enough knowledge to determine its state. Similarly, in the proposed process some metrics are calculated both intra- and inter- logs with the purpose of analyzing the overall behavior of the external partners.

The main novelty of the proposed process is the use of empirical knowledge collected during the service provision to guarantee compliance with both individual rules and the overall contract. Unlike current methods based on a runtime policy verification, I have aimed at an aposteriori verification which is framed in a semi-trustworthiness approach. As rules cannot be directly verified since the behaviors of the external partner are not observable, more concrete information is required to verify the compliance with the policy. Therefore, it can be argued that partners do not entirely trust the behaviors of external organizations and ask for evidence, but partners trust the sent evidence. Indeed, in case of mistakes, the defendant can be released of liability if he/she proves that he/she behaves correctly and that the mistake comes from the external partner. Trust is good, but evidence is better.

6.2 Thesis Perspectives

As a research work is never completely concluded, I will present some insights of the possibilities which were open during the course of this thesis, some of them are simple improvements to the proposed approach which can be done

in the short-term, while others have a larger scope and complexity requiring a long-term approach.

- A clear improvement which can be made to the present work is the integration of the proposed controllability rule model with an ontological representation. Indeed, in Chapter 4, an analysis was made to argue why the SWRL model, which combines ontologies and rules, is not suitable to the problem identified in this dissertation. However, it is still possible to create a language based on the proposed model to the definition of business contractual policies. For this, it will be necessary to modify the parser and the inference engine to hold the proposed representation. Namely, a Model-driven Engineering (MDE) approach may be a powerful tool for the development of the proposed language.

- A second perspective regarding the modeling process consists in improving the expressiveness of controllability rules to support alternative behaviors. It has been discussed in Chapter 4 that one possible solution would be to include an "else" block in the rule structure. Similarly, the process of auditing should also be consequently modified to consider that the non-compliance with an obligation has no negative impact for the evaluation of the actor or the contract if other obligations has been successfully accomplished.

- Throughout this thesis, the assumption of having one single provider and one single client was made. This was purposely assumed to avoid the negotiation process, which was considered as out of the scope of this work. An interesting perspective is related to consider multi-party contracts which include the negotiation process. In this sense, the use of intelligent agents can prove useful, considering that the proposed ontology can be used to improve the communication ability of agents and that the log can be used as the knowledge that the agent has of its environment.

- Regarding the contract evaluation, it has been argued in Chapter 4 that the proposed process does not replace existing approaches; conversely, it aims at other kind of guarantees which cannot be verified in runtime through the traditional monitoring techniques. Consequently, the integration of runtime techniques with heuristic (aposteriori) methods can be investigated. Indeed, the runtime monitoring process can effectively support the verification of some environmental conditions defined in the context.

- Finally, at the end of Chapter 4, it has been mentioned that other heuristic techniques, and more concretely, Artificial Neural Networks (ANN) can be used to improve the processing of the knowledge collected during the service provision. During the development of the proposed process, I have identified that the contract compliance can be seen as a classification problem in which some heuristic weighted inputs are used to compute the state of the contract. In this case, the knowledge recorded in the log can be used to train the neural network through the adaptation of the weight of each input according to the history archive of contract states.

Conclusion (Français)

De façon générale, la fourniture d'un service concerne la livraison d'un produit tangible ou intangible à un consommateur. Le paradigme orienté services a permis d'informatiser ce processus en rendant les services accessibles à un large nombre d'utilisateurs, en faisant abstraction de l'homogénéité technologique et en automatisant les tâches telles que l'authentification, l'autorisation, la sélection des fournisseurs et la vérification des paramètres de performance.

S'appuyant sur l'état de l'art, il a été défendu dans cette thèse l'importance d'établir de façon explicite les termes régissant une prestation de service. Tout d'abord, il s'agit d'aligner les attentes du client avec le service réellement fourni. Deuxièmement, de créer les bases d'une relation de confiance entre le client et le fournisseur, dans laquelle chaque partie sait ce qui se fait au cours de la prestation de services. Troisièmement, d'évaluer la conformité des engagements. En particulier, les organisations ont compris la nécessité de préciser clairement ce que chaque partenaire peut, ne peut pas, devrait ou doit faire au cours de la prestation de services. En effet, dans le contexte B2B, une "dépendance d'affaire" est générée de manière implicite, causée par le fait que certains actifs sont partagés entre les clients et les fournisseurs, et que la manière dont ils sont utilisés par le partenaire externe peut causer des dommages organisationnels tels que la perte de clients, des pénalités financières, la perte de réputation et des poursuites judiciaires. Même si, de manière générale, un actif fait référence à toute ressource organisationnelle, ce travail a été particulièrement axé sur les actifs partagés, car l'enjeu de cette approche est qu'ils se déplacent d'un domaine organisationnel à un autre, et que les activités commerciales dans lesquelles l'actif est impliqué induisent son utilisation.

C'est en fait une question complexe qui a émergé dans les technologies axées sur les services parce que les organismes sont sensibles à la prévention des dommages qui peuvent être causés par les mauvais comportements du partenaire externe. Cependant, il n'y a aucun moyen de savoir si l'actif a été correctement utilisé, quand il a quitté le périmètre du contrôle et que les processus de l'organisation externe sont considérés comme des boîtes noires. Cette thèse aborde le problème de la perte de contrôle dans l'utilisation des actifs partagés au cours d'une prestation de services, ce qui a été défini comme le problème de contrôlabilité. Même si ce problème concerne le B2B, ainsi que le B2C, nous avons supposé un scénario de B2B car cela permet de prendre en considération le cas le plus général dans lequel les clients et les fournisseurs veulent garder le contrôle sur leurs actifs respectifs. Il a été également montré que les approches actuelles visant des garanties de service non fonctionnels ne satisfont pas la nécessité de représenter les exigences opérationnelles sur l'utilisation prévue des

actifs.

Au cours des dernières années, il y a eu un intérêt croissant de la communauté scientifique pour la recherche sur le comportement humain comme élément pertinent de la sécurité. Tenant compte du fait que, d'une part, même si les partenaires collaborent au cours de la prestation de services, ils peuvent encore avoir des objectifs opposés, et d'autre part que les personnes peuvent parfois avoir un comportement irrationnel, considérer l'utilisation des actifs comme une facette du comportement du partenaire n'est pas un problème trivial. Tout au long de cette thèse, nous avons défendu et démontré qu'il est non seulement possible de modéliser le comportement des partenaires en utilisant des stratégies de contrôlabilité, mais aussi qu'il est possible d'utiliser les connaissances disponibles recueillies au cours de la prestation de services, afin de vérifier la conformité avec la politique modélisée. Ainsi, il a été proposé que l'expressivité des contrats actuels interprétables par la machine soit étendue aux exigences de contrôlabilité. En suivant, les contributions de la thèse sont reprises et analysées à la lumière des objectifs. Puis, les perspectives de ce travail sont décrites.

Objectifs et Contributions

Définir une représentation interprétable par la machine capable de formaliser des garanties concernant la fourniture d'un service.

Ayant à l'esprit l'objectif général de modéliser la contrôlabilité dans le cadre des exigences non-fonctionnelles qui régissent une prestation de service, l'état de l'art du Chapitre 3 a été axé sur les approches actuelles visant à représenter des garanties lors de la fourniture d'un service sous une forme interprétable par la machine. La recherche dans ce domaine a mis en évidence deux approches principales, à savoir : les certificats et les SLA, qui ont été analysés pour déterminer s'ils pouvaient être adaptés pour répondre aux besoins en termes de contrôlabilité. Bien que l'adaptation soit possible, la nature des changements serait telle que le principe de ces approches doit être repensé. En effet, les documents écrits en anglais réglementant la fourniture d'un service ont été pris comme base pour identifier les besoins en termes de représentation, et plusieurs résultats nous ont amenés à conclure qu'une approche alternative était nécessaire. Tout d'abord, les organisations définissent des contraintes sur une gamme d'actifs plus large que les seules données. Ensuite, les exigences opérationnelles relatives à l'utilisation des actifs sont exprimées en termes d'activités organisationnelles à gros grains au lieu d'actions à grains fins. Enfin, étant donné que les actifs et les activités sont représentés à l'aide d'un vocabulaire d'affaires, une sémantique claire est nécessaire afin d'éviter des erreurs d'interprétation. Par conséquent, non seulement l'expressivité des approches actuelles aurait besoin d'être étendue, mais également le mécanisme de vérification puisque les exigences de contrôlabilité abordent des activités qui ne sont pas observables ou mesurables au cours de l'exécution.

Ainsi, nous avons proposé au Chapitre 3 la définition de contrats sémantiques interprétables par la machine, ce qui permet de représenter les exigences opérationnelles réissant la fourniture d'un service, dont la modélisation est le résultat d'un processus de marquage fait sur un ensemble de contrats de service de cas

réels écrit en anglais. Il a été démontré que le formalisme $\mathcal{SROIQ}(\mathcal{D})$ offre l'expressivité requise pour définir les concepts spécialisés de notre domaine. L'avantage d'un tel formalisme est son expressivité robuste et sa décidabilité, ce qui le rend apte à une représentation interprétable par la machine. En particulier, le langage OWL et sa syntaxe concrète XML ont été utilisés pour représenter le contrat sémantique.

La représentation proposée aboutit à un modèle générique de contrats de service. Bien que certaines autres représentations sémantiques aient été proposées dans le domaine des services, dans le périmètre de nos connaissances, la relation contractuelle entre les clients et les fournisseurs n'a pas été étudiée. Elle consiste en la première contribution de cette thèse.

Proposer une définition formelle de la contrôlabilité dans le cadre des exigences non-fonctionnelle de l'entreprise, régissant la relation entre les clients et fournisseurs de services.

La contrôlabilité a été définie comme le contrôle de l'utilisation des actifs. Dans la littérature, les méthodes de contrôle d'utilisation ont déjà été proposés par les modèles UCON; cependant, dans ces travaux l'"utilisation" est définie en fonction de la continuité de l'accès. Cela signifie une évaluation en continue dans laquelle les droits accordés peuvent être révoqués lors de l'utilisation d'objets si certaines propriétés ne sont pas plus valables. De la même manière que pour nous, [68] considère la contrôlabilité d'un point de vue organisationnel, et plus particulièrement l'utilisation des actifs est associée à la gestion des risques. Néanmoins, les auteurs de ce travail ne proposent pas de définition formelle de la contrôlabilité et elle est seulement présentée de façon abstraite comme une nouvelle propriété en sécurité.

La deuxième contribution de cette thèse est donc une formalisation sémantique du contrôle de l'utilisation des actifs, où le vocabulaire de contrôlabilité vient enrichir le modèle de contrat. Un tel vocabulaire définit les acteurs de contrôlabilité ainsi qu'une taxonomie de l'actif qui est conforme à la norme ISO/IEC 27005, et la définition des activités commerciales en termes d'opérations effectuées sur ces actifs.

Proposer un modèle pour les exigences de contrôlabilité qui soit en mesure de représenter le comportement attendu d'un partenaire externe en ce qui concerne l'utilisation des actifs.

Dû au fait que le premier objectif était de représenter les exigences opérationnelles dans un contrat de service, il est clair que la définition des exigences de contrôlabilité devient une partie intégrante des conditions contractuelles en contraignant les clients et les fournisseurs à l'accomplissement de certains engagements. L'état de l'art présenté dans le Chapitre 4 a été axé sur les approches actuelles pour représenter les politiques, en particulier celles impliquant plusieurs organisations. Trois principales conclusions ont conduit à proposer un modèle spécifique pour représenter les politiques de contrôlabilité. Tout d'abord, contrairement à notre approche, la plupart des politiques inter-organisationnelles actuelles supposent que l'actif reste dans le domaine du contrôle, où certains droits sont accordés à des partenaires externes. Deuxièmement, les

objets et les actions définies dans les politiques examinées sont limitées en ce qui concerne les besoins de l'organisation en termes de contrôlabilité. Enfin, les politiques sont destinées à être appliquées lors de l'exécution, ce qu'il n'est pas possible de faire avec les règles de contrôlabilité puisque les actions du partenaire externe sont inconnues, et par conséquent, non monitorables afin de préserver la logique interne de chaque organisation.

Considérant que la manière avec laquelle chaque organisation utilise les actifs définit son comportement, dans le chapitre 4 nous proposons un modèle pour les règles de contrôlabilité composées de quatre éléments principaux, à savoir : un sujet, qui définit un acteur et certains attributs particuliers; un comportement, qui définit une activité commerciale et ses conditions; un contexte, qui définit le système et les conditions environnementales; et enfin, une modalité, qui indique si le comportement attendu correspond à une permission, une recommandation, une obligation ou une interdiction. En particulier, la signification des éléments décrits dans ces composants est tirée du modèle sémantique présenté au Chapitre 3. Il vise à surmonter d'éventuelles erreurs d'interprétation entre les parties contractuelles en raison de l'utilisation d'un vocabulaire métier. Outre la définition structurelle des règles, certaines métadonnées complètent leur description sémantique pour représenter la pertinence de l'usage défini, qui est modélisée comme le poids de la règle; de même, puisque la règle ne peut pas être directement vérifiée, des preuves interprétables par la machine sont définies pour démontrer la conformité avec le comportement attendu; enfin, le but de la règle est modélisé.

La représentation des clauses contractuelles interprétables par la machine sur la base du modèle proposé contribue à étendre l'expressivité des SLA actuels aux garanties de services sur le comportement attendu des acteurs. Ces garanties sont en mesure d'exprimer des exigences plus complexes associées aux besoins de l'entreprise au-delà de la sécurité et des garanties d'exécution.

Vérifier la conformité avec les règles de contrôlabilité, afin d'évaluer la performance des acteurs du service et la qualité du service fourni.

Dans les chapitres 3 et 4 la partie modèle de la méthode proposée a été présentée. Elle était composée de deux éléments. Le premier est la modélisation de la sémantique des contrats de services, incluant un vocabulaire de contrôlabilité, et le deuxième, la modélisation des règles de contrôlabilité dans le cadre des conditions contractuelles. Dans le chapitre 5, a été démontré la nécessité de vérifier la conformité avec les règles contractuelles, ce qui constitue une étape importante dans le cadre d'une approche de gouvernance. En effet, décrire de manière explicite les besoins de contrôlabilité ne garantit pas leur conformité, ni tire parti des avantages d'une représentation interprétable par la machine des contrats par rapport à des représentations en texte brut. Le principal défi de la vérification de la politique est que le prix à payer pour étendre l'expressivité des accords de service est de représenter les activités et les attributs qui ne peuvent pas être vérifiables ou mesurable au cours de l'exécution, ce qui rend difficile l'évaluation de leur conformité avec des techniques de surveillance traditionnelles. Dans le chapitre 5, un processus visant à vérifier la conformité avec le contrat est proposé. Il utilise les métadonnées, et en particulier, le poids de chaque règle et les preuves, pour créer un journal qui est utilisé comme une

base de connaissances pour déduire l'état de chaque règle, et par conséquent, de déterminer si le contrat n'a pas été respecté, respecté ou s'il n'y a pas assez d'éléments pour déterminer son état. De même, dans le processus proposé certains paramètres sont calculés, tant intra-journal que inter-journal, dans le but d'analyser le comportement global des partenaires extérieurs.

La principale nouveauté de la démarche proposée est l'utilisation des connaissances empiriques recueillies au cours de la prestation de services afin de garantir la conformité avec les règles individuelles et l'ensemble du contrat. Contrairement aux méthodes actuelles basées sur une vérification de la politique au cours de l'exécution, nous avons ciblé une vérification a posteriori dans le cadre d'une approche de semi-confiance. Cela signifie, en raison du fait que les règles ne peuvent pas être vérifiées directement car les comportements du partenaire externe ne sont pas observables, que des informations plus concrètes sont nécessaires pour vérifier la conformité avec la politique. Par conséquent, on peut affirmer que les partenaires n'ont pas entièrement confiance dans les comportements des organismes externes, vu qu'ils demandent des preuves ; mais, d'autre part, le partenaire fait confiance aux preuves reÃ§ues. En effet, en cas de défaut, le partenaire mis en cause peut être libéré de responsabilité s'il prouve qu'il s'est comporté correctement et que la faute venait du partenaire externe. La confiance est une bonne chose, mais avoir des preuves est encore mieux.

Perspectives de thèse

Comme un travail de recherche n'est jamais complètement terminé, dans ce qui suit, nous présentons un aperçu des perspectives qui s'ouvrent dans la lignée de cette thèse, certains d'entre elles sont de simples améliorations à l'approche proposée qui peuvent être faites à court terme, alors que d'autres ont une portée plus grande et leur complexité suppose une vision à long terme.

1. Une première amélioration concerne le développement du modèle de règles de contrôlabilité proposé avec une représentation ontologique. En effet, dans le chapitre 4, une analyse a été faite pour argumenter pourquoi le modèle SWRL, qui combine les ontologies et les règles, ne convient pas à notre problème. Cependant, il est possible d'envisager la création d'un langage basé sur le modèle proposé pour la définition des politiques contractuelles. Pour cela, il sera nécessaire de modifier l'analyseur et le moteur d'inférence pour maintenir la représentation proposée. À cet égard, une approche IDM - Ingénierie Dirigée par les Modèles peut être un outil puissant pour le développement du langage proposé.

2. Une seconde perspective est liée à améliorer l'expressivité des règles de contrôlabilité pour permettre de modéliser des comportements alternatifs. Il a été mentionné dans le chapitre 5 qu'une solution possible serait d'inclure un bloc "else" dans la structure de la règle. Pour cela, le processus de vérification doit également être modifié en conséquence. Dès lors, le non-respect d'une obligation n'a pas d'impact négatif pour l'évaluation de l'acteur ou du contrat si une autre obligation a été accomplie avec succès.

3. Tout au long de cette thèse, l'hypothèse d'avoir un seul fournisseur et un seul client a été faite. Cela pour éviter le processus de négociation,

qui a été considéré comme hors du périmètre scientifique de ce travail. Une perspective intéressante est d'envisager des contrats multi-parties qui conduisent à un processus de négociation. Pour cela, l'utilisation d'agents intelligents peut s'avérer utile, étant donné que l'ontologie proposée peut être utilisée pour améliorer la capacité de communication des agents et que le journal peut être utilisé comme la connaissance que l'agent a de son environnement.

4. En ce qui concerne l'évaluation du contrat, il a été défendu dans le Chapitre 4 que le processus proposé ne remplace pas les approches existantes, mais qu'il vise d'autres types de garanties qui ne peuvent pas être vérifiées à l'exécution en utilisant des techniques de surveillance traditionnelles. Par conséquent, il peut être envisagé une recherche autour de l'intégration des méthodes de vérification en temps réel et des méthodes de vérification heuristiques (a posteriori). En effet, la surveillance en temps réel peut soutenir efficacement la vérification de certaines conditions environnementales définies dans le contexte.

5. Enfin, à la fin de Chapitre 4, il a été mentionné que d'autres techniques heuristiques, et plus concrètement, les réseaux de neurones artificiels (ANN) peuvent être utilisées pour améliorer le traitement des connaissances recueillies au cours de la prestation de services. Lors de l'élaboration du processus proposé, il a été mis en évidence que le respect du contrat peut être considéré comme un problème de classification dans laquelle certaines entrées pondérées sont utilisées pour calculer l'état du contrat. Dans ce cas, la connaissance enregistrée dans le journal peut être utilisée pour former le réseau de neurones en adaptant le poids de chaque entrée selon l'historique des états du contrat.

Appendices

Appendix A

DL Expressiveness

A.1 Expressiveness of DL-Family of Languages

	$\mathcal{ALC}$	$\mathcal{ALCF}$	$\mathcal{ALCQIO}$	$\mathcal{SOIQ}$	$\mathcal{SHOIQ}$	$\mathcal{ROIQ}$	$\mathcal{SHOIN}$
Atomic concept	√	√	√	√	√	√	√
Atomic concept negation	√	√	√	√	√	√	√
Conjunction of classes	√	√	√	√	√	√	√
Disjunction of classes	√	√	√	√	√	√	√
Existential quantification	√	√	√	√	√	√	√
Universal quantification	√	√	√	√	√	√	√
Functional property		√					
Unqualified number restriction							√
Qualified number restriction			√	√	√	√	
Nominals			√	√	√	√	√
Inverse role			√	√	√	√	√
Role Transitivity				√	√	√	√
Role Hierarchy					√	√	√
Complex role Inclusion						√	

Table A.1: Expressiveness Power of DL

A.2 Complexity of the DL-Family of Languages

Expressiveness	**Complexity**
$\mathcal{ALC}$	PSpace-Complete
$\mathcal{ALCF}$	PSpace-Complete
$\mathcal{ALCQIO}$	NExpTime-Complete
$\mathcal{SOIQ}$	NExpTime-Complete
$\mathcal{SHOIQ}$	NExpTime-Complete
$\mathcal{ROIQ}$	NExpTime-Hard, Decidable
$\mathcal{SHOIN}$	NExpTime-Complete

References

[1] Giuseppe Aceto, Alessio Botta, Walter De Donato, and Antonio Pescapè. Cloud monitoring: A survey. *Computer Networks*, 57(9):2093–2115, 2013.

[2] M. F. Al-Jaberi and A. Zainal. Data integrity and privacy model in cloud computing. In *Biometrics and Security Technologies (ISBAST), 2014 International Symposium on*, pages 280–284, Aug 2014.

[3] G.H. Alférez, V. Pelechano, R. Mazo, C. Salinesi, and D. Diaz. Dynamic adaptation of service compositions with variability models. *Journal of Systems and Software*, 91:24 – 47, 2014.

[4] A. Almutairi and F. Siewe. Formal specification of ca-ucon model using cca. In *Science and Information Conference (SAI), 2013*, pages 369–375, Oct 2013.

[5] Alain Andrieux, Karl Czajkowski, Asit Dan, Kate Keahey, Heiko Ludwig, Toshiyuki Nakata, Jim Pruyne, John Rofrano, Steve Tuecke, and Ming Xu. Web Services Agreement Specification (WS-Agreement). Technical report, Global Grid Forum, Grid Resource Allocation Agreement Protocol (GRAAP) WG, September 2005.

[6] M. Anisetti, C. A. Ardagna, and E. Damiani. Security certification of composite services: A test-based approach. In *Web Services (ICWS), 2013 IEEE 20th International Conference on*, pages 475–482, June 2013.

[7] M. Anisetti, C.A. Ardagna, E. Damiani, P.A. Bonatti, M. Faella, C. Galdi, and L. Sauro. E-auctions for multi-cloud service provisioning. In *Services Computing (SCC), 2014 IEEE International Conference on*, pages 35–42, June 2014.

[8] Marco Anisetti, Claudio Agostino Ardagna, Michele Bezzi, Ernesto Damiani, and Antonino Sabetta. Machine-readable privacy certificates for services. In *On the Move to Meaningful Internet Systems: OTM 2013 Conferences - Confederated International Conferences: CoopIS, DOA-Trusted Cloud, and ODBASE 2013, Graz, Austria, September 9-13, 2013. Proceedings*, pages 434–450, 2013.

[9] Marco Anisetti, Claudio Agostino Ardagna, Ernesto Damiani, and Francesco Saonara. A test-based security certification scheme for web services. *TWEB*, 7(2):5, 2013.

[10] Claudio A Ardagna, Ravi Jhawar, and Vincenzo Piuri. Dependability certification of services: a model-based approach. *Computing*, 97(1):51–78, 2015.

[11] Axelos. *ITIL Practitioner Guidance*. Stationery Office Limited, 2016.

[12] S. Ayed. *Secure workflow deployment in multi-organizational environments*. PhD thesis, LUSSI - Dépt. Logique des Usages, Sciences Sociales et de l'Information (Institut Mines-Télécom-Télécom Bretagne-UBL), 2009. Th. doct. : Informatique, Institut Mines-Télécom-Télécom Bretagne-UBL, 2009.

[13] Alistair Barros, Marlon Dumas, and Arthur H. M. ter Hofstede. *Service Interaction Patterns*, pages 302–318. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[14] Abhijit Belapurkar, Anirban Chakrabarti, Harigopal Ponnapalli, Niranjan Varadarajan, Srinivas Padmanabhuni, and Srikanth Sundarrajan. *Distributed Systems Security - Issues, Processes and Solutions*, pages 1–307. John Wiley & Sons, Ltd, 2009.

[15] Michele Bezzi and Slim Trabelsi. *Data Usage Control in the Future Internet Cloud*, pages 223–231. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[16] Todd Biske. *SOA Governance*. Packt Pub., 2008.

[17] Alex Borgida and Luciano Serafini. *Distributed Description Logics: Assimilating Information from Peer Sources*, pages 153–184. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[18] S. Boukhedouma, M. Oussalah, Z. Alimazighi, and D. Tamzalit. Adaptation patterns for service based inter-organizational workflows. In *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*, pages 1–10, May 2013.

[19] Emiliano Casalicchio and Luca Silvestri. Mechanisms for sla provisioning in cloud-based service providers. *Computer Networks*, 57(3):795–810, 2013.

[20] Paolo Ceravolo, Cristiano Fugazza, and Marcello Leida. Modeling semantics of business rules. In *2007 Inaugural IEEE-IES Digital EcoSystems and Technologies Conference*, pages 171–176. IEEE, 2007.

[21] Namyoun Choi, Il-Yeol Song, and Hyoil Han. A survey on ontology mapping. *ACM Sigmod Record*, 35(3):34–41, 2006.

[22] S. Cimato, E. Damiani, F. Zavatarelli, and R. Menicocci. Towards the certification of cloud services. In *2013 IEEE Ninth World Congress on Services*, pages 92–97, June 2013.

[23] V. Cisternino, A. Corallo, G. Elia, and C. Fugazza. Business rules for semantics-aware business modelling: overview and open issues. *Int. J. Web Eng. Technol.*, 5(1):104–131, May 2009.

[24] The World Wide Web Consortium. Xml signature syntax and processing. W3C Recommendation V 1.1, W3C, 2013.

[25] Microsoft Corporation. Improving Web Services Security: Scenarios and Implementation Guidance for WCF. https://msdn.microsoft.com/en-us/library/ff650794.aspx, 2009. [Online; accessed 29-June-2016].

[26] E. Damiani, S. De Capitani di Vimercati, C. Fugazza, and P. Samarati. Extending context descriptions in semantics-aware access control. In *Proceedings of the Second International Conference on Information Systems Security*, ICISS'06, pages 162–176, Berlin, Heidelberg, 2006. Springer-Verlag.

[27] Ernesto Damiani, Sabrina De Capitani di Vimercati, Cristiano Fugazza, and Pierangela Samarati. *Extending Policy Languages to the Semantic Web*, pages 330–343. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[28] Ernesto Damiani, Sabrina De Capitani di Vimercati, Cristiano Fugazza, and Pierangela Samarati. Modality conflicts in semantics aware access control. In *Proceedings of the 6th International Conference on Web Engineering*, ICWE '06, pages 249–256, New York, NY, USA, 2006. ACM.

[29] Sergio de Cesare and Guido L. Geerts. *Toward a Perdurantist Ontology of Contracts*, pages 85–96. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[30] Ministerio Secretaria General de la Presidencia de Chile. Ley 19628 - ley sobre proteccion de la vida privada. http://www.leychile.cl/Navegar?idNorma=141599, 2012.

[31] D. D. Downs, J. R. Rub, K. C. Kung, and C. S. Jordan. Issues in discretionary access control. In *Security and Privacy, 1985 IEEE Symposium on*, pages 208–208, April 1985.

[32] Vincent C Emeakaroha, Marco AS Netto, Rodrigo N Calheiros, Ivona Brandic, Rajkumar Buyya, and César AF De Rose. Towards autonomic detection of sla violations in cloud infrastructures. *Future Generation Computer Systems*, 28(7):1017–1029, 2012.

[33] Jeremy Epstein, Scott Matsumoto, and Gary McGraw. Software security and soa: Danger, will robinson! *IEEE Security & Privacy*, 4(1):80–83, 2006.

[34] European Parliament and the Council of the European Union. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. *Official Journal of the European Union*, L 281:0031–0050, 1995.

[35] European Parliament and the Council of the European Union. Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications). *Official Journal of the European Union*, L 201:0037–0047, 2002.

[36] European Parliament and the Council of the European Union. Directive 2010/87/EU on standard contractual clauses for the transfer of personal data to processors established in third countries under Directive 95/46/EC of the

European Parliament and of the Council. *Official Journal of the European Union*, L 318:0032–0035, 2010.

[37] European Parliament and the Council of the European Union. Directive (EU) 2016/680 of the European Parliament and of the Council. *Official Journal of the European Union*, L 119:89–131, 2016.

[38] Dieter Fensel, Holger Lausen, Axel Polleres, Jos de Bruijn, Michael Stollberg, Dumitru Roman, and John Domingue. *Enabling Semantic Web Services: The Web Service Modeling Ontology*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[39] Stefan Fenz and Andreas Ekelhart. Formalizing information security knowledge. In *Proceedings of the 4th international Symposium on information, Computer, and Communications Security*, pages 183–194. ACM, 2009.

[40] David Ferraiolo, Janet Cugini, and D Richard Kuhn. Role-based access control (rbac): Features and motivations. In *Proceedings of 11th annual computer security application conference*, pages 241–48, 1995.

[41] Howard Foster and George Spanoudakis. Smart: a workbench for reporting the monitorability of services from slas. In *Proceedings of the 3rd International Workshop on Principles of Engineering Service-Oriented Systems*, pages 36–42. ACM, 2011.

[42] Vivek Gaur, P Dhyani, and OP Rishi. A multi-objective optimization of cloud based sla-violation prediction and adaptation. *International Journal of Information Technology and Computer Science*, 8(6):60–65, 2016.

[43] N. M. Gonzalez, M. A. T. Rojas, M. V. M. da Silva, F. Redigolo, T. C. M. d. B. Carvalho, C. C. Miers, M. Näslund, and A. S. Ahmed. A framework for authentication and authorization credentials in cloud computing. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pages 509–516, July 2013.

[44] Niels Grewe. A generic reification strategy for n-ary relations in dl. In *OBML 2010 Workshop Proceedings*, 2010.

[45] The Open Group. Service-oriented architecture ontology, version 2.0. The Open Group Technical Standard Reference C144, The Open Group, 2014.

[46] Andreas Haeberlen. A case for the accountable cloud. *ACM SIGOPS Operating Systems Review*, 44(2):52–57, 2010.

[47] David Hannah, Emily Treen, Leyland Pitt, and Pierre Berthon. But you promised! managing consumers' psychological contracts. *Business Horizons*, 59(4):363 – 368, 2016.

[48] C. Hoertnagl, J.F. Riordan, and D. Bourges-Waldegg. Semantic digital signatures. `http://www.google.com/patents/US20090319794`, December 24 2009. US Patent App. 12/141,245.

[49] G. Hohpe and B. Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. A Martin Fowler signature book. Addison-Wesley, 2004.

[50] K. Huang, M. Xian, S. Fu, and J. Liu. Securing the cloud storage audit service: defending against frame and collude attacks of third party auditor. *IET Communications*, 8(12):2106–2113, August 2014.

[51] W. Hussain, F. K. Hussain, and O. K. Hussain. Comparative analysis of consumer profile-based methods to predict sla violation. In *Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on*, pages 1–8, Aug 2015.

[52] innoQ. Web Services Standards as of Q1 2007. `https://www.innoq.com/resources/ws-standards-poster/`, 2008. [Online; accessed 05-July-2016].

[53] International Working Group on Data Protection in Telecommunications. Working Paper on Cloud Computing - Privacy and data protection issues - Sopot Memorandum. *51st Meeting, 23-24 April 2012, Sopot (Poland)*, 2012.

[54] ISACA. *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT.* COBIT® 5. ISACA, 2012.

[55] ISO/IEC. ISO/IEC 27002:2005 - Information technology – Security techniques – Code of practice for information security management. Technical report, ISO/IEC, 2005.

[56] ISO/IEC. *ISO/IEC 27005. Information security risk management.* ISO/IEC, 2011.

[57] ISO/IEC. Information Technology - Security Techniques - Information Security Management Systems: Overview and Vocabulary. Technical report, ISO/IEC, 2016.

[58] Gloria Elena Jaramillo, Manuel Munier, and Philippe Aniorte. Information security in business intelligence based on cloud: A survey of key issues and the premises of a proposal. In *WOSIS 2013 - Proceedings of the 10th International Workshop on Security in Information Systems, In conjunction with ICEIS 2013, Angers, France, 4-7 July, 2013*, 2013.

[59] H. Jayasree and A. Damodaram. Anonymity and accountability in web based transactions. *Advanced Computing : An International Journal*, 3(2):171–182, 2012.

[60] Vandana Kabilan, Paul Johannesson, and Dickson M. Rugaimukamu. *Business Contract Obligation Monitoring through Use of Multi Tier Contract Ontology*, pages 690–702. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[61] A. A. E. Kalam, R. E. Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miege, C. Saurel, and G. Trouessin. Organization based access control. In *Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*, pages 120–131, June 2003.

[62] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. *The knowledge engineering review*, 18(01):1–31, 2003.

[63] Paul Karaenke and Stefan Kirn. Service level agreements: An evaluation from a business application perspective. In *Proceedings of eChallenges*, 2007.

[64] K. T. Kearney, F. Torelli, and C. Kotsokalis. Sla*: An abstract syntax for service level agreements. In *2010 11th IEEE/ACM International Conference on Grid Computing*, pages 217–224, Oct 2010.

[65] Tong Ka Kent. *Developing web services with Apache CXF and Axis2*. Lulu. com, 2010.

[66] Markus Krötzsch. OWL 2 Profiles: An introduction to lightweight ontology languages. In Thomas Eiter and Thomas Krennwallner, editors, *Proceedings of the 8th Reasoning Web Summer School, Vienna, Austria, September 3–8 2012*, volume 7487 of *LNCS*, pages 112–183. Springer, 2012.

[67] V. Lalanne, M. Munier, and A. Gabillon. Information security risk management in a world of services. In *Social Computing (SocialCom), 2013 International Conference on*, pages 586–593, Sept 2013.

[68] Vincent Lalanne. *Gestion des risques dans les architectures orientées services*. PhD thesis, Université de Pau et des Pays de l'Adour, 2013.

[69] D. D. Lamanna, J. Skene, and W. Emmerich. Slang: A language for defining service level agreements. In *Distributed Computing Systems, 2003. FTDCS 2003. Proceedings. The Ninth IEEE Workshop on Future Trends of*, pages 100–106, May 2003.

[70] Farah Layouni and Yann Pollet. Fi-orbac: a model of access control for federated identity platform. In *IADIS International Conference Information Systems Barcelona, Spain*, 2009.

[71] D. S. Linthicum. *Cloud Computing and SOA Convergence in Your Enterprise: A Step-by-Step Guide*. Addison-Wesley Information Technology Series. Pearson Education, 2009.

[72] D. S. Linthicum. How the cloud has modernized SOA. `http://www.infoworld.com/`, 2016. [Online; accessed 29-June-2016].

[73] Paulo Lopes Cardoso, Henrique; Leitão and Eugenio Oliveira. An approach to inter-organizational workflow management in an electronic institution. In *Proceedings of the 11th IFAC Symposium on Information Control Problems in Manufacturing*, 2006.

[74] L. Lowis and R. Accorsi. On a classification approach for soa vulnerabilities. In *Computer Software and Applications Conference, 2009. COMPSAC '09. 33rd Annual IEEE International*, volume 2, pages 439–444, July 2009.

[75] Heiko Ludwig, Alexander Keller, Asit Dan, Richard P. King, and Richard Franck. Web Service Level Agreement (WSLA) Language Specification, v1.0, January 2003.

[76] Xiaofeng Luo, Lin Li, and Wanbo Luo. A contextual usage control model. *Tehnicki vjesnik/Technical Gazette*, 21(1), 2014.

[77] Azadeh Mellat, Naser Nematbakhsh, Ahmad Farahi, and Farhad Mardukhi. Suitability of uml state machines for modeling choreography of services. *International Journal of Web & Semantic Technology*, 2(4):33 – 53, 2011.

[78] C. Müller, M. Oriol, X. Franch, J. Marco, M. Resinas, A. Ruiz-Cortes, and M. Rodriguez. Comprehensive explanation of sla violations at runtime. *IEEE Transactions on Services Computing*, 7(2):168–183, April 2014.

[79] M. Munier, V. Lalanne, and M. Ricarde. Self-protecting documents for cloud storage security. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 1231–1238, June 2012.

[80] Manuel Munier, Vincent Lalanne, Pierre-Yves Ardoy, and Magali Ricarde. *Legal Issues About Metadata Data Privacy vs Information Security*, pages 162–177. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[81] Manuel Munier, Vincent Lalanne, Pierre-Yves Ardoy, and Magali Ricarde. Métadonnées et Aspects Juridiques: Vie Privée vs Sécurité de l'Information. In *9ème Conférence sur la Sécurité des Architectures Réseaux et des Systèmes d'Information (SARSSI'2014)*, pages 65–76, Saint-Germain-Au-Mont-d'Or, France, May 2014.

[82] S. Nepal, J. Zic, and S. Chen. Wsla+: Web service level agreement language for collaborations. In *Services Computing, 2008. SCC '08. IEEE International Conference on*, volume 2, pages 485–488, July 2008.

[83] Zijian NI and RONG Lili. Using relation triangle for guiding reification of n-ary relations in owl. *Journal of Network & Information Security*, 4(3):199–206, 2013.

[84] David Nuñez, Carmen Fernandez-Gago, Siani Pearson, and Massimo Felici. A metamodel for measuring accountability attributes in the cloud. In *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, volume 1, pages 355–362. IEEE, 2013.

[85] Office of the Privacy Commissioner for Personal Data of Hong Kong. An overview of the major provisions of the personal data-privacy and amendment, 2012.

[86] ICO Information Commissioners's Office. Actions we've taken - Enforcement. `https://ico.org.uk/action-weve-taken/enforcement/`, 2016. [Online; accessed 05-June-2016].

[87] Laima Paliulioniene. On description of contracts and agreements in the context of soa. *Computational Science and Techniques*, 1(2):171–183, 2013.

[88] Adrian Paschke. Rbsla - a declarative rule-based service level agreement language based on ruleml. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, volume 2, pages 308–314, Nov 2005.

[89] Siani Pearson, Vasilis Tountopoulos, Daniele Catteddu, Mario Südholt, Refik Molva, Christoph Reich, Simone Fischer-Hübner, Christopher Millard, Volkmar Lotz, Martin Gilje Jaatun, et al. Accountability for cloud and other future internet services. In *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, pages 629–632. IEEE, 2012.

[90] A. Pretschner, F. Schütz, C. Schaefer, and T. Walter. Policy evolution in distributed usage control. *Electronic Notes in Theoretical Computer Science*, 244:109 – 123, 2009.

[91] Eric Pulier and Hugh Taylor. *Understanding Enterprise SOA*. Manning Publications Co., Greenwich, CT, USA, 2005.

[92] Sumalatha M. R., Hemalathaa S., Monika R., and Ahila C. Towards secure audit services for outsourced data in cloud. In *Recent Trends in Information Technology (ICRTIT), 2014 International Conference on*, pages 1–6, April 2014.

[93] Simona Ramanauskaitė, Dmitrij Olifer, Nikolaj Goranin, and Antanas Čenys. Security ontology for adaptive mapping of security standards. *International Journal of Computers, Communications & Control (IJCCC)*, 8(6):813–825, 2013.

[94] Christopher Rees. Who owns our data? *Computer Law & Security Review*, 30(1):75 – 79, 2014.

[95] Thomas Reiter, Manuel Wimmer, and Horst Kargl. Towards a runtime model based on colored petri-nets for the execution of model transformations. In *3rd Workshop on Models and Aspects@ ECOOP*, volume 7, pages 19–23, 2007.

[96] Christoph Ringelstein and Steffen Staab. Logging in distributed workflows. In *Proceedings of the 2007 International Conference on Privacy Enforcement and Accountability with Semantics-Volume 320*, pages 19–30. CEUR-WS. org, 2007.

[97] Denise M. Rousseau. New hire perceptions of their own and their employer's obligations: A study of psychological contracts. *Journal of Organizational Behavior*, 11(5):389–400, 1990.

[98] Denise M. Rousseau and Snehal A. Tijoriwala. Assessing psychological contracts: issues, alternatives and measures. *Journal of Organizational Behavior*, 19(S1):679–695, 1998.

[99] Mustapha Ben Saidi and Abderrahim Marzouk. Multi-trust_orbac: Access control model for multi-organizational critical systems migrated to the cloud. *International Journal of Soft Computing and Engineering (IJSCE)*, 3(2), 2013.

[100] S. J. Samuel and S. J. Jenitha. Enhanced security and authentication mechanism in cloud transactions using hmac. In *Computational Intelligence and Computing Research (ICCIC), 2014 IEEE International Conference on*, pages 1–4, Dec 2014.

[101] R. S. Sandhu and P. Samarati. Access control: principle and practice. *IEEE Communications Magazine*, 32(9):40–48, Sept 1994.

[102] Paula Severi, José Fiadeiro, and David Ekserdjian. Guiding the representation of n-ary relations in ontologies through aggregation, generalisation and participation. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(2):83–98, 2011.

[103] Malaysia. Jabatan Standard. *Risk Management: Vocabulary (ISO Guide 73:2009,IDT)*. Malaysian standard. Department of Standards Malaysia, 2010.

[104] United States. *Uniting and Strengthening America by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism (USA PATRIOT ACT) Act of 2001*. U.S. Government Printing Office, 2001.

[105] Clare Sullivan. Protecting digital identity in the cloud: Regulating cross border data disclosure. *Computer Law & Security Review*, 30(2):137 – 152, 2014.

[106] Validation PhD Thesis. OWL Code of the Semantic Model. `https://drive.google.com/open?id=0B_VwLS1AeIMmclQxUU1yeW1VWE0`, 2016.

[107] Vladimir Tosic, Bernard Pagurek, Kruti Patel, Babak Esfandiari, and Wei Ma. Management applications of the web service offerings language (wsol). *Information Systems*, 30(7):564 – 586, 2005. The 15th International Conference on Advanced Information Systems Engineering (CAiSE 2003)The 15th International Conference on Advanced Information Systems Engineering (CAiSE 2003).

[108] Vladimir Tosic, Kruti Patel, and Bernard Pagurek. *WSOL - Web Service Offerings Language*, pages 57–67. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[109] Andries van Dijk. Contracting workflows and protocol patterns. In WilM.P. van der Aalst and Mathias Weske, editors, *Business Process Management*, volume 2678 of *Lecture Notes in Computer Science*, pages 152–167. Springer Berlin Heidelberg, 2003.

[110] Wattana Viriyasitavat. Multi-criteria selection for services selection in service workflow. *Journal of Industrial Information Integration*, 1:20 – 25, 2016.

[111] W3C. Test Suite Status. `https://www.w3.org/2007/OWL/wiki/Test_Suite_Status`, 2009. [Online; accessed 13-August-2016].

[112] Hongda Wang, Jianchun Xing, Qiliang Yang, Ping Wang, Xuewei Zhang, and Deshuai Han. Optimal control based regression test selection for service-oriented workflow applications. *Journal of Systems and Software*, pages 1 – 15, 2016.

[113] Manuel Wimmer, Horst Kargl, Martina Seidl, Michael Strommer, and Thomas Reiter. Integrating ontologies with car-mappings. In *First International Workshop on Semantic Technology Adoption in Business (STAB'07), Vienna, Austria*. Citeseer, 2007.

[114] Linlin Wu and Rajkumar Buyya. Service level agreement (sla) in utility computing systems. *IGI Global*, 2012.

[115] El Senado y Camara de Diputados de la Nacion Argentina. *Ley 25.326 sobre la Proteccion de los Datos Personales*, 2000.

[116] Yalan Yan, Jinlong Zhang, and Mi Yan. Ontology modeling for contract: Using owl to express semantic relations. In *2006 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06)*, pages 409–412. IEEE, 2006.

[117] Jinhui Yao, Shiping Chen, Chen Wang, David Levy, and John Zic. Accountability as a service for the cloud. In *Services Computing (SCC), 2010 IEEE International Conference on*, pages 81–88. IEEE, 2010.

[118] Veruska Zamborlini and Giancarlo Guizzardi. An ontologically-founded reification approach for representing temporally changing information in owl. In *11th International Symposium on Logical Formalizations of Commonsense Reasoning*, 2013.

[119] Evgeny Zolin. Complexity of reasoning in Description Logics. `http://www.cs.man.ac.uk/~ezolin/dl/`, 2013.