République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Blida 1
Faculté des sciences
Département d'informatique

Mémoire de fin d'études en vue de l'obtention du diplôme de
MASTER EN INFORMATIQUE

Option : système informatique et réseau

THÈME :

Description de services dans le multicloud

Réalisé par
Meharzi cherifa

Devant le jury
Mme CHERFA Imane (Présidente)
M. DOUGA Yassine(Examinateur)
Mme. MANCER Yasmine (Promotrice)

Promotion : 2021/2022

Remerciement

Je veux d'abord remercier Allah, qui a donné la force et la capacité pour réaliser ce travail.

Je remercie également ma promotrice, Mme Mancer, qui a guidé et aidé tout au long de mon travail.

Je remercie sincèrement mes chers parents, ma famille et tous mes amis.

A tous ces intervenants, je vous présente mes remerciements, mon respect et ma gratitude.

Enfin, je tiens à remercier les membres du jury qui ont accepté d'évaluer mon travail.

Résumé

Aujourd'hui, le Cloud Computing est une des technologies les plus adoptées, elle offre une panoplie d'avantages et de services avec des architectures rentables. Il est vu comme étant une interconnexion et une coopération de ressources de l'informatique, située dans des entités et structures internes, externes ou mixtes dont les modes d'accès sont basés sur les protocoles et standard Internet. Cependant, les défis auxquels le Cloud Computing doit faire face sont également en constante augmentation et la description de services Cloud en est un des plus complexes.

Les langages , les modèles , les ontologies , les normes ont tous été utilisé pour décrire les services du Cloud Computing , cependant aucun d'entre eux n'est capable de décrire les fonctionnalités spéciales de tous les types de services Cloud (laas , SaaS , PaaS) , du point de vue technique , commercial , opérationnel et sémantique. De plus, la non standardisation de la description des services Cloud constitue un grand obstacle dans les processus de découverte, de sélection et de composition pour les entreprises et les particuliers.

Dans ce travail, nous proposons une solution destinée aux fournisseurs qui souhaitent mettre à disposition leurs services, mais aussi aux clients qui veulent bénéficier de ces services. Notre but est d'arriver à une description générale regroupant le maximum des notions de Cloud et prenant en compte tous les aspects (sémantique, opérationnel, business et technique), pour tous les types de service Cloud (laaS, PaaS, SaaS). Pour cela, nous avons réalisé des adaptations entre les langages de description les plus utilisés pour la description des services Cloud. L'aspect sémantique a été complété en utilisant des ontologies.

Pour l'expérimentation de la solution proposée, nous avons réalisé une application java qui permet la création de services et l'utilisation de leurs descriptions pour la découverte et la sélection.

Mots clés : Cloud Computing, multicloud, Description, Sélection, Transformation de modèle, interopérabilité.

Abstract

Today, Cloud Computing is one of the most adopted technologies, it offers a panoply of advantages and services with profitable architectures. It is seen as an interconnection and cooperation of IT resources, located in internal, external or mixed entities and structures whose access modes are based on Internet protocols and standards. However, the challenges that Cloud Computing must face are also constantly increasing and the description of Cloud services is one of the most complex.

Languages, models, ontologies, standards have all been used to describe Cloud Computing services, however none of them is able to describe the special features of all types of Cloud services (laas, SaaS, PaaS), from a technical, commercial, operational and semantic point of view. In addition, the non-standardization of the description of Cloud services constitutes a great obstacle in the processes of discovery, selection and composition for companies and individuals.

In this work, we propose a solution intended for suppliers who wish to make their services available, but also for customers who want to benefit from these services. Our goal is to arrive at a general description bringing together as many Cloud concepts as possible and taking into account all aspects (semantic, operational, business and technical), for all types of Cloud service (laaS, PaaS, SaaS). For this, we have made adaptations between the most used description languages for the description of Cloud services. The semantic aspect was completed using ontologies.

For the experimentation of the proposed solution, we realized a java application which allows the creation of services and the use of their descriptions for the discovery and the selection.

Keywords : Cloud Computing, multicloud, Description, Selection, Model transformation, interoperability.

SOMMAIRE

LIST DES FIGURES

LISTE DES TABLEAUX

LISTE DES ACRONYMES ET ABRÉVIATIONS

AHP Analytic Hierarchy Process

ASP Application Service Provider

ATL Atlas Transformation Language

CACM ommunications of the Association for Computing Machinery

DBaas Database as a Service

DSaaS The data-Storage-as-a-Service

EC2 Elastic Compute Cloud

GCSD Generale cloud service description

IaaS Infrastructure as a Service

MCDM Multiple Criteria Decision Analysis

Naas Network as a Service

NIST National Institute of Standards and Technology

OS Operating System

OWL-S Web Ontology Language for Web Services

PaaS Platform as a Service

REST REpresentational State Transfer

SaaS Software as a Service

SLA Service Level agreement

SOAP Simple Object Access Protocol

TIC Technologies de l'Information et de la Communication

USDL Unified Service Description Language

VM machines virtuelles

W3C World Wide Web Consortium

WaaS Workplace as a Service

WSDL Web Services Description Language

WSMO Web Service Modeling Ontology

XaaS Anything as a Service

XML eXtensible Markup Language

INTRODUCTION GÉNÉRALE

Le cloud computing a connu une énorme diffusion et popularité entre les fournisseurs et les clients. La raison étant que le cloud computing offre une solution à la demande et un accès à un pool de ressources informatiques partagées avec une disponibilité et une évolutivité élevées, en mode ”pay-as-you-go”. Malgré les nombreux avantages du cloud computing, elle a des obstacles tels que : l'hétérogénéité des architectures et des API, le grand nombre de fournisseurs et de technologies, l'absence de standard, le problème du « Vendor Lock - In ‘ » etc. De plus,de nombreux problèmes apparaissent aux utilisateurs et compliquent la tâche de découverte de services , car les fournisseurs utilisent des techniques distinctes pour la description de leurs services , (telles que les langages , les normes , les ontologies , ou les modèles , etc.) . Cette diversité de techniques conduit les fournisseurs de services à fournir différentes interfaces et d'utiliser différents protocoles pour l'accès aux services. Ce manque de représentation standard de services de Cloud empêche l'interopérabilité entre les services et renforce le problème de verrouillage des fournisseurs.

Problématique

Les langages de description existant ne répondent pas à tous les aspects de description, par exemple, le langage WSDL prend en charge l'aspect technique mais ne couvre pas les aspects business et sémantique. Les chercheurs considèrent USDL (Unified Service Description Language) comme un langage qui couvre la description de service à partir de trois aspects (technique , opérationnel et business) , mais il ne prend pas en compte l'aspect sémantique et n'est pas conçu pour le domaine du Cloud Computing.De plus, les langages existants ne

contiennent pas les notions générale pour le cloud et s'il existe un description générale comment se fait la transformation des langages à cette description pour pouvoir réaliser l'interopérabilité entre les Clouds. .

Objectifs

Les objectifs visés à travers ce travail sont les suivants :

1. Etude de la description de service Cloud Computing.
2. Etude comparative des solutions existantes pour la description de service cloud.
3. Etude comparative des solutions existantes pour la transformation des langages vers le langage de description.
4. Définir une solution pour la description des services Cloud , en considérant les aspects (technique , opérationnel , business et sémantique) .
5. Définir une solution pour rendre les langages de description adaptés au cloud compuing.
6. Définir une solution pour la transformation des langages vers le langage de description et la transformation inverse.
7. Validation de la solution proposée par une expérimentation à travers une application Java EE.

Organisation du mémoire

Le travail décrit dans ce mémoire est organisé en quatre chapitres.

- **Chapitre I :** Le premier chapitre présente les notions de base du Cloud Computing telles que la définition du Cloud Computing, les modèles de livraison et de déploiement des services de Cloud Computing, principales caractéristiques du Cloud Computing , les fondations du Cloud Computing , architecture du Cloud Computing, les avantages, les inconvénients et les défis rencontrés dans ce domaine et l'évolution du Cloud Computing .
- **Chapitre II :** Le deuxième chapitre présente des généralités et des définitions avec un état de l'art sur la description des services Cloud et les solutions existantes pour la description et la transformation.
- **Chapitre III :** Modélisation de la solution proposée et ses différents concepts.
- **Chapitre IV :** Expérimentation de la solution proposée et réalisation d'une application java qui permet la création et la sélection des services cloud.

- Enfin, nous terminerons par une conclusion générale et quelques perspectives pour les travaux futurs.

CHAPITRE I

GÉNÉRALITÉS SUR LE CLOUD COMPUTING

I.1 Introduction

Le développement des technologies de l'information et de la communication (TIC) a donné naissance à un nouveau paradigme : le cloud computing. Bien que ce paradigme soit relativement jeune, il continue de prendre de l'ampleur et d'intéresser les entreprises et la communauté scientifique. En effet, les perspectives de cette nouvelle vision de l'informatique sont nombreuses et, si les éventuels obstacles à son adoption ne peuvent être ignorés, elle est souvent occultée par ses avantages.

Dans ce chapitre, nous introduisons les différents concepts du Cloud Computing. Nous commençons par présenter les diverses définitions attribuées au Cloud Computing, et nous déterminons les notions de base, les caractéristiques, types de déploiement, types de services, ete.).Enfin et avant de conclure, nous soulignons l'évolution du Cloud Computing.

I.2 Définition du Cloud Computing

Le Cloud Computing est un nuage de services et de données. Plus précisément, c'est un paradigme, et à ce titre il est difficile de lui donner une définition exacte et de dire avec certitude s'il s'agit ou non de Cloud. Beaucoup de gens sont confus au sujet de ce que le Cloud Computing. En gros, il décrit les ressources hautement évolutives fournies comme un service externe via l'Internet sur un paiement à l'utilisation. Cloud Computing peut être défini comme un modèle spécialisé de calcul distribué, qui est configuré dynamiquement et livré à la demande. Ce nouveau paradigme massivement évolutif est différent des réseaux traditionnels.

Définition selon National Institute of Standards and Technology (NIST [1]) :

"Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable Computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [1]

1. *https ://www.nist.gov/* est une agence du département du Commerce des États-Unis. Son but est de promouvoir l'économie en développant des technologies, la métrologie et des normes de concert avec l'industrie.

I.3 Types des services dans l'environnement de Cloud Computing

Le modèle de service permet de décrire différentes caractéristiques d'un environnement Cloud relativement aux types de services offerts et, de ce fait, à la façon dont un client peut accéder aux ressources Cloud.

Le NIST [1] répertorie trois principaux modèles de service Cloud : IaaS (Infrastructure as a Service), le PaaS (Platform as a Service) et le SaaS (Software as a Service).

Dans cette section, nous décrivons les trois types de service précédemment cités. Ceux ci sont les types de service de base du Cloud Computing. Cependant, certaines entreprises proposent des modèles de services plus ciblés et plus restreints en termes de fonctionnalités. Nous passons également en revue quelques-uns de ces modèles dans cette section.

1. **Software as a Service (SaaS)**

   Dans une couche SaaS, le fournisseur propose comme service une application qui est disponible sur le Cloud. Le client peut alors consommer ce service à la demande, sans être dans l'obligation d'en acheter la licence afin de l'installer sur ses machines de travail. Il pourra en bénéficier à distance, sans devoir connaitre son emplacement précis. Un fournisseur de software as a service peut exploiter des services de type platform as a service, qui peut lui-même se servir de infrastructure as a service.[1]

2. **Plateform as a Service(PaaS)**

   Dans ce type de service, situé juste au-dessous du précédent, le système d'exploitation et les outils d'infrastructure sont sous la responsabilité du fournisseur. Le consommateur a le contrôle des applications et peut ajouter ses propres outils.

   La couche PaaS offre aux clients un environnement de type midd- leware. C'est une plate-forme de développement permettant la création et l'exploitation de logiciels. Cet environnement contient les bases de données, les logiciels serveur, l'infrastructure serveur et le stockage.[1]

3. **Infrastructure as a Service (IaaS)**

   C'est le service de plus bas niveau. Il consiste à offrir un accès à un pare informatique virtuelles . Des machines virtnelles sur lesquelles le consommateur peut installer un système d'exploitation et des applications.[1]

4. **Autres types de services Cloud**

   Bien que trois types de services de cloud computing (laaS, PaaS et SaaS) soient à la base de la distinction des types de services, le terme « as a Service » a été utilisé ailleurs pour caractériser les services et les ressources de cloud computing. Chaque nouvelle abréviation peut être considérée comme un sous-ensemble d'un ou plusieurs des trois types de base. Par conséquent,aujourd'hui, nous parlons de XaaS (Anything as a Service). Dans les abréviations existantes, nous citons :

   (a) **NaaS (Network as a Service) :**Un modèle de cloud computing où les locataires peuvent accéder à d'autres ressources informatiques qui coexistent avec des commutateurs et des routeurs. Il s'agit d'un modèle de fourniture virtuelle de services réseau via des modèles de service par abonnement ou « pay as you go ». Avec NaaS, tout ce dont le client a besoin est un ordinateur avec connexion Internet, qui est connecté au portail NaaS fourni par le fournisseur NaaS/cloud. NaaS simplifie l'architecture du réseau grâce à la virtualisation. Cisco [2], Aryaka [3] tet BigSwitch [4] sont des réseaux bien connus en tant que fournisseurs de services.[5]

   (b) **WaaS (Workplace as a Service) :** Le Workplace as service est un environnement de travail virtuel disponible partout quelque soit l'appareil utilisé, il permet aux utilisateurs finaux professionnels un accès sécurisé, permanent et en tout lieu à leurs applications et données d'entreprise, par exemple, Econocom workplace as a service, workspace as a service de colt [8]

   (c) **DBaaS (Database as a Service) :**Base de données en tant que service, il s'agit d'un modèle de service de cloud computing qui permet aux utilisateurs d'accéder aux bases de données sans avoir besoin de configurer du matériel physique, d'installer des logiciels ou d'effectuer des performances. Le fournisseur de services est responsable

2. *https ://www.cisco.com/*
3. Aryaka is a SD-WAN company that provides software-defined net work connectivity and application delivery to globally distributed enterprises.*https ://www.aryaka.com/*
4. BigSwitch Networks is a company that is a paert of Arista Neworks. It was founded in 2010 by Douglas Murra

de toutes les táches de gestion et de maintenance, donc tout ce que l'utilisateur ou le propriétaire de l'application doit faire est d'utiliser la base de données.[5]

(d) **DSaaS(The data-Storage-as-a-Service) :** fournit le stockage que le consommateur veut utilisé et les besoins en bande passante pour le stockage.[4]

Le tableau I.1 suivant donne des exemples d'offres de services Clouds existants sur le marché des leaders mondiaux.

SaaS	PaaS	IaaS
• **Google** offre Google Apps (messagerie et bureautique) • **Sales Force CRM** (Customer Relation-ship Management)	• **Microsoft** offre Azur • **Google** offre Google App Engine	• **Amazon** offres EC2 et AWS • **Microsoft** offre Azur

TABLE I.1 – Les grands acteurs mondiaux du Cloud Computing [6]

I.4 Modèles de déploiement

Le modèle de déploiement permet de décrire un environnement Cloud sur trois points principaux : son propriétaire, sa taille et le type d'accès qu'il permet. Il existe quatre différents-modèles de déploiement que nous présentons ci-dessous.

1. **Cloud privé**

   Le cloud privé (également appelé cloud interne) est un terme marketing désignant une architecture informatique propriétaire qui fournit des services gérés à un nombre limité de personnes derrière un pare-feu. Généralement, les clouds privés sont implémentés dans les centres de données d'entreprise et contrôlés par des ressources internes.

   Le cloud privé conserve les données d'entreprise dans des ressources sous la tutelle des lois et des contrats de l'organisation. [9]

2. **Cloud public**

   Le cloud public (ou cloud externe) est un modèle standard de cloud computing, dans lequel les fournisseurs de services fournissent des ressources telles que des applications et du stockage au public sur Internet. Le service cloud peut être gratuit ou fourni sur un modèle de paiement à l'utilisation. L'un des principaux avantages de ce type de cloud est qu'il est simple à mettre en place et peu coûteux, car les coûts du matériel, des applications et de la bande passante sont à la charge du fournisseur. Les clouds externes sont également connus pour leur évolutivité afin de répondre à la demande. [9]

3. **Cloud hybride**

   Pour profiter des les avantages des deux approches précédentes , de nouveaux modèles d'exécution ont été d'enveloppés pour combiner les Clouds publics et privés dans une solution unifiée, c'est les Clouds hybrides. Des applications avec des préoccupations d'ordre juridique, réglementaire ou d'un service important pour les renseignements peuvent être dirigées vers un Cloud privé. D'autres applications avec des conditions moins rigoureuses de normalisation ou d'un service moins strict peuvent s'appuyer sur une infrastructure de Cloud public. La mise en oeuvre d'um modèle hybride nécessite une coordination supplémentaire entre les secteurs privé et public du système de gestion des services. Ceci implique un outil de gestion fédéré de politique, une sécurité fédérée une gestion des biens d'information, un contrôle coordonné d'approvisionnement et un système unifié de surveillance. [9]

   Par exemple, une organisation peut utiliser un service de Cloud public, comme le Cloud d'Amazon Elastic Compute (EC2) pour l'informatique générale et stocke les données des

clients dans son propre centre de données.

4. **Cloud communautaire**

   L'infrastructure d'un Cloud communautaire est partagée par plusieurs organisations indépendantes ayant des intérêts communs. L'infrastructure peut être gérée par les organisations membres ou par un tiers. L'infrastructure peut être située, soit au sein des dites organisations, soit chez un fournisseur de services.[7]

I.5 Principales caractéristiques du Cloud Computing

En règle générale, un service, une solution ou un environnement d'exécution doit répondre à une série de caractéristiques pour être considéré comme un cloud computing. Parmi ces caractéristiques, certaines sont considérées comme des caractéristiques de base.C'est pourquoi seules 5 caractéristiques sont mises en avant par la société NIST. [10]

I.5.1 Accès libre à la demande

Le cloud computing permet aux clients d'utiliser des ressources informatiques comme services, telles que les serveurs, stockage ou puissance de calcul, plateformes de développement, à travers le réseau, selon leurs besoins, de manière simple et flexible, sans la nécessité d'une interac tion manuelle avec les fournisseurs de services. Le cloud est donc un système autonome, visant à construire un système informatique capable de s'autogérer et de s'adapter aux changements in- ternes et externes. Matériel, logiciels et données cloud peuvent être automatiquement reconfiguré, orchestré et intégré dans une seule image qui sera fournie aux utilisateurs.[11]

I.5.2 Large accés réseau

Les services fournis par le cloud aux utilisateurs doivent être disponibles sur le réseau et disponibles via des mécanismes standard qui favorisent l'utilisation de la plate-forme hétérogène, tels que les ordinateurs portables, les postes de travail, les tablettes et les téléphones portables. [11]

I.5.3 Ressources partagées

Le cloud computing est un nouveau modèle informatique basé sur le modèle économique où les services et les ressources sont agrégés et fournis aux consommateurs sur le modèle multi-

locataire.Contrairement aux modèles traditionnels, les ressources physiques ou virtuelles sont allouées dynamiquement en fonction de la demande.[11]

I.5.4 Élasticité rapide

Les ressources informatiques peuvent être rapidement augmentées et diminuées selon les besoins. De plus, ces ressources peuvent être complètement libérées pour d'autres utilisations lorsqu'elles ne sont plus nécessaires. Contrairement aux infrastructures traditionnelles qui ont des capacités fixes et limitées, le Cloud Computing permet de redimensionner massivement les ressources (bande passante, espace de stockage, etc.). Cette élasticité dérive des services Cloud évolutifs pour satisfaire les demandes croissantes des utilisateurs. Cette évolutivité doit se faire d'une façon automatique et en cours d'exécution.[11]

I.5.5 Service mesuré

Le Cloud assure la transparence entre le fournisseur et le consommateur du service, en fournissant des outils permettant aux consommateurs du Cloud de contrôler et de superviser l'utilisation de leurs ressources.[11]

En plus de ces cinq caractéristiques, il y a d'autres caractéristiques dont nous citons les plus pertinentes :[11]

- Paiement à l'usage
- Basé sur la virtualisation
- Basé sur le SLA (Service Level agreement)
- Simplicité, flexibilité, fiabilité et tolérance aux pannes

I.6 Définition de SLA(Service Level Agreement)

Un accord de niveau de service (Service Level Agreement en anglais) définit le niveau de service attendu par un client d'un fournisseur, définissant les paramètres selon lesquels ce service est mesuré, et les remèdes ou pénalités, le cas échéant, si les niveaux de service convenus ne sont pas respectés.

Habituellement, les accords de niveau de service sont conclus entre des entreprises et des

fournisseurs externes, mais ils peuvent également se faire entre deux services au sein d'une entreprise.

Destiné à baliser le bon déroulement d'un projet, le SLA est un élément incontournable de tout contrat signé avec un prestataire informatique.[75]

I.7 Éléments technologiques du Cloud Computing

Un environnement Cloud Computing ne peut être défini sans un ensemble d'éléments qui sont :

I.7.1 La virtualisation

La virtualisation se définit comme l'ensemble des techniques matérielles et/ou logicielles qui permettent de faire fonctionner sur une seule machine, plusieurs systèmes d'exploitation, appelées machines virtuelles (VM), ou encore OS invité.[12]
La virtualisation des serveurs permet une plus grande modularité dans la répartition des charges et la reconfiguration des serveurs en cas d'évolution ou de défaillance momentanée.

I.7.2 Centre de calcul (Datacenter)

Un centre de traitement de données est un site physique, sur lequel se trouvent regroupés des équipements constituants le système d'information de l'entreprise (mainframes, serveurs, baies de stockage, équipements réseaux et de télécommunications, etc.). Il peut être interne et/ou externe à l'entreprise, exploité ou non avec le soutien de prestataires. Il comprend en général un contrôle sur l'environnement (climatisation, système de prévention contre l'incendie, etc.), une alimentation d'urgence et redondante, ainsi qu'une sécurité physique élevée. Cette infrastructure peut être propre à une entreprise et utilisé par elle seule à des fins commerciales. Ainsi, des particuliers ou des entreprises peurvent venir y stocker leurs données suivant des modalités bien définies. [12]

I.7.3 Plateforme collaborative

Une plate-forme de travail collaborative est un espace de travail virtuel. C'est un site qui centralise tous les outils liés à la conduite d'un projet et les met à disposition des acteurs . L'objectif du travail collaboratif est de faciliter et d'optimiser la communication entre les individus

dans le cadre du travail ou d'une tâche.

Les plates-formes collaboratives intègrent généralement les éléments suivants : des outils informatiques ; des guides ou méthodes de travail en groupe pour améliorer la communication, la production et la coordination ; un service de messagerie ; un système de partage des ressources et des fichiers ; des outils de type forum ou pages de discussions ; un annuaire des profils des utilisateurs, des groupes, par projet ou par thématique et un calendrier. [12]

I.8 Architecture du Cloud Computing

I.8.1 Architecture globale du cloud computing

En faisant abstraction de détails de plus haut niveau, l'architecture globale du cloud computing comporte essentiellement[15] :

- **Des clients**
  personne, entreprise,groupe qui accèdent aux différent services offerts par le cloud.
- **Des services**
  Différent niveaux de services et données sont gérés par des fournisseurs afin de les offrir à la demande des clients du cloud.
- **Un réseau**
  Intermédiaire entre le client et le fournisseur qui permet de transiter les services(chemin que les services entreprend), c'est le réseau Internet.
- **Des fournisseurs**
  Ce sont des entités chez lesquelles on alloue les services cloud.
- **Des serveurs**
  Ou sont stockés les services cloud, ils sont éparpiés partout dans le monde et constituent le coeur hardware du cloud computing.

I.8.2 Architecture de référence du cloud computing selon NIST

La (figure I.1) présente un aperçu de l'architecture de référence du Cloud Computing de NIST [11], qui identifie les acteurs majeurs, leurs activités et leurs fonctions dans le Cloud Computing. Le diagramme représente une architecture générique de haut niveau et est destiné à faciliter la compréhension des exigences, des utilisations, des caractéristiques et des normes

de l'informatique en nuage.

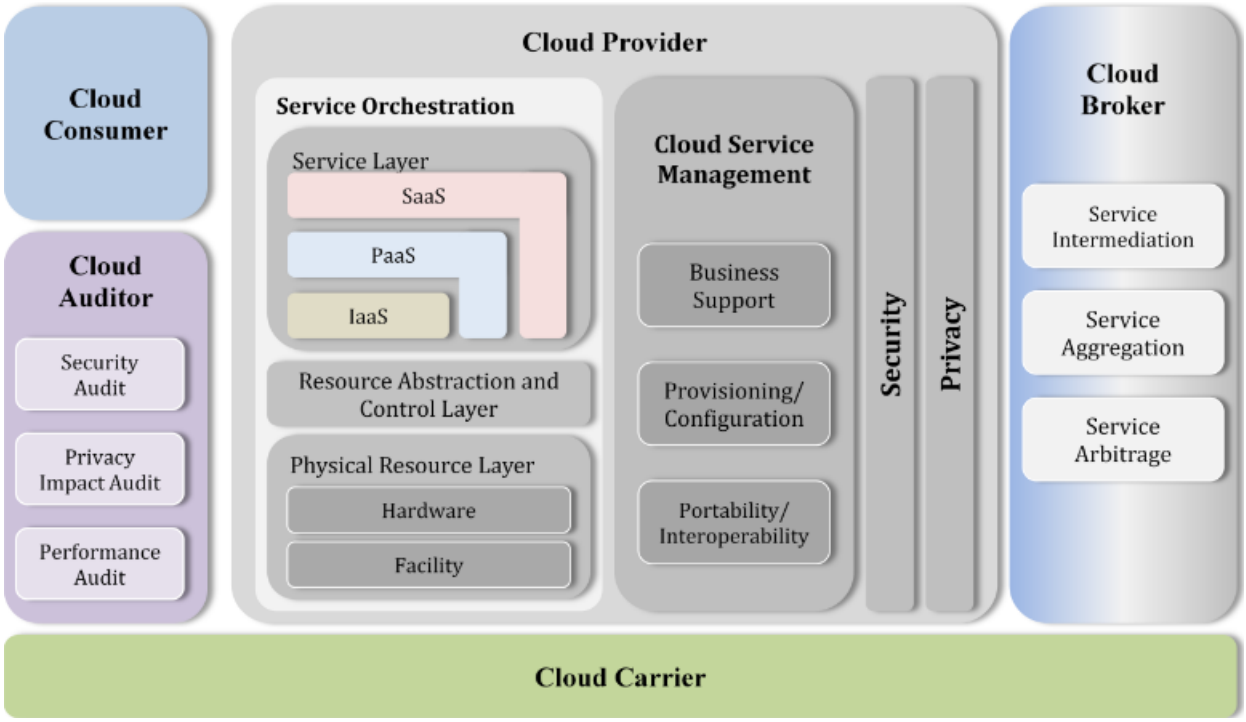

FIGURE I.1 – Modèle conceptuel de référence du Cloud Computing[10]

I.8.3 Les acteus du Cloud Computing

Selon NIST, l'écosystème du Cloud Computing est composé principalement de cinq acteurs majeurs (Cloud Provider, Cloud Consumer, Cloud Carrier, Cloud Broker, Cloud Auditor) :

- **Cloud Provider : Le fournisseur des ressources Cloud Computing.**
  Il est responsable de fournir un service Cloud Computing qui satisfait ses caractéristiques tout en respectant les Service Level Agreements(SLAs) établies avec les autres acteurs (en particulier le Cloud Consumer). Le Cloud Provider a comme activité l'allocation, l'orchestration et la gestion des ressources qu'il offre tout en assurant le bon niveau de sécurité.[14]
- **Cloud Consumer : L'utilisateur des ressources Cloud Computing.**
  Cet utilisateur peut être un utilisateur final ou un développeur selon le type du service Cloud alloué. Il peut être une personne, un groupe de personnes, petites ou moyennes entreprises, les multinationales ou les gouvernements. [14]
- **Cloud Carrier ou Network Provider :**
  Le fournisseur de réseau est l'intermédiaire qui assure principalement la connectivité entre les ressources Cloud Computing et la liaison entre les acteurs de l'écosystème Cloud Computing (en particulier entre le Cloud Provider et le Cloud Consumer). Cet utilisateur peut jouer un simple rôle d'acheminements des paquets, comme il peut jouer un rôle plus important en offrant des fonctionnalités avancées dans le réseau. Ces fonctionnalités sont basées sur des SLAs établies avec les autres acteurs de l'écosystème. [14]
- **Cloud Broker :**
  Le courtier Cloud est un intermédiaire qui négocie la relation entre les Cloud Providers et les Cloud Consumers. Il peut offrir de nouveaux services qui simplifient les tâches de gestion du Cloud Consumer. Ce dernier peut demander les ressources Cloud Computing auprès du Cloud Broker au lieu du Cloud Provider directement. Pour récapituler, le Cloud Broker peut assurer l'orchestration, l'agrégation et l'arbitrage des services Cloud Computing [14]
- **Cloud Auditor :**
  L'auditeur Cloud s'occupe de la vérification et l'audition des services Cloud Computing. Il évalue les services offerts par les Cloud Providers, Cloud Carriers et Cloud Brokers du point de vue performances et sécurité. Le but principal est de vérifier que les fournisseurs respectent bien les SLAs qu'ils proposent.[14]

I.9 Les avantages et les inconvénients du Cloud Computing

Le marché des services Clouds connait une croissance fulgurante ces dernières années, prouvant ainsi qu'il ne s'agit pas juste d'un boom médiatique mais une vraie technologic dans le domaine des apptications distribuées et aussi pour la délivrance de services. Cela est dû principalement à sa politique d'utilisation qui offre de nombreux avantages. Toutefois, l'adoption du Cloud Computing se heurte à plusieurs obstacles. [17]

I.9.1 Avantages

• **Réduction des coûts de gestion :** avec le Cloud les entreprises n'ont plus à se soucier de la gestion des ressources ou du personnel nécessaire à la supervision de leurs plateformes.

• **Système anti désastre :** la récupération des données et des applications après un désastre (séisme par exemple est gérée par un backend qui stocke et relance le système à nouveau pour assurer une disponibilité permanente des services Cloud.

• **Mobilité et accès facile :** en stockant nos données et en déployant nos applications sur le Cloud, l'accès à ces derniers devient seulement une question de connexion Internet.

• **Réduction des coûts d'utilisation :** le modèle économique du Cloud permet aux clients de réduire et de contrôler leurs dépenses, puisque ils ne payent que ce qu'ils utilisent comme ressources Cloud.

I.9.2 Inconvénients

• **Conformité réglementaire :** lors du transfert des données du client vers le fournisseur du service Cloud, le client est seul responsable de l'intégrité et de la sécurité des données.

• **Dépendance des services :** un client n'a pas la possibilité de changer le type de services à consommer chez un fournisseur donné.

• **Vie privée et lois de confidentialités :** les lois qui régissent la préservation des données privées est différente d'un pays à un autre (par exemple, les USA n'ont pas les mêmes lois que l'union européenne), ce qui peut mettre les clients en situation de confusion par rapport aux

services Cloud à utiliser, et cela peut limiter le transfert de données d'une ressource Cloud vers une antre qui se trouve dans un autre pays.

• **Récupération de données :** le fait de segmenter les taches et les données stockées par les clients et ensuite les éparpillés sur l'infrastructure Cloud, rend leur récupération et par la suite leur rassemblement très difficile.

• **Identification des clients :** avec l'utilisation croissante du Cloud et l'utilisation multilocation de ces ressources, il devient de plus en plus difficile d'identifier par qui et de quel endroit les données ont été modifiées.

• **Stockage de données :** le stockage physique des données dans le Cloud est effectué par les fournisseurs des services ce qui limite la manipulation de ces dernier par les clients.

I.10 Evolution du Cloud Computing

L'adoption massive récente des services cloud a augmenté la demande des clients cherchant des services de plus en plus polyvalents. Les exigences deviennent de plus en plus complexes, nécessitant la collaboration de plusieurs clouds, et peut-être même de différents fournisseurs, pour répondre aux exigences des clients. Cette évolution a donné naissance à de nouveaux domaines tels que le multi-cloud et l'inter-cloud. Pour bien comprendre les concepts de multi-cloud et de l'inter-cloud nous devons d'abord comprendre les concepts d'interopérabilité et de portabilité dans les clouds informatiques.

• **L'interopérabilité :**Pour tout système informatique, l'interopérabilité est la capacité de deux ou plusieurs systèmes ou applications à échanger des informations et à utiliser les informations échangées entre eux. Dans le Cloud Computing, l'interopérabilité est la capacité des systèmes clients à interagir avec les services cloud ou la capacité des services cloud à interagir avec d'autres services cloud et à échanger des informations selon des méthodes prescrites pour obtenir des résultats prévisibles.[20]]

• **La portabilité :**Cela inclut le déplacement de données et/ou d'applications d'un système à un autre et leur maintien disponibles ou exécutables. La portabilité des données cloud fait référence à la capacité de déplacer facilement des données d'un service cloud à un autre sans

avoir à ressaisir les données. La portabilité des applications cloud est la possibilité de migrer des applications d'un service cloud vers un autre service cloud ou entre l'environnement client et le service cloud.

I.10.1 Multi-Cloud

Une approche multi-cloud fait référence à l'utilisation de plusieurs clouds. Il est indépendant du fournisseur de cloud et utilise des ressources de différents cloud. Contrairement à une fédération de Clouds, un environnement multi-Cloud n'implique pas l'interconnexion des bénévoles et le partage des infrastructures des fournisseurs. Les clients ou leurs représentants sont directement responsables de la gestion de l'approvisionnement des ressources et de la planification. [19]

I.10.2 Inter-Cloud

L'inter-Cloud est le concept de réseaux de Clouds connectés, comprenant des Clouds publics privés et hybrides. Il intègre un certain nombre d'activités technologiques regroupées pour améliorer l'interopérabilité et la portabilité entre les réseaux en nuage.[18]
Le concept d'inter-cloud est encore vague, et les chercheurs sont divisés sur la définition de base. Contrairement à une stratégie multi-cloud, une stratégie inter-cloud nécessite un accord préalable entre les différents fournisseurs de cloud.

I.11 Défis du cloud computing

L'utilisation du Cloud Computing apporte plusieurs avantages soit au niveau des utilisateurs, soit au niveau des fournisseurs de services. Malgré cela, il existe un certain nombre de défis liés à son utilisation. On énumère quelques uns de ces défis comme suit :

- **La sécurité :** La sécurité informatique joue un rôle important dans le déploiement du Cloud Computing.Elle est le plus grand défi à relever par les responsables informatiques qui souhaitent adopter des solutions et des services hébergés dans le Cloud. [21]
- **L'interopérabilité :** veut dire qu'une application de Cloud Computing est capable de s'adapter et de collaborer avec d'autres applications indépendantes de Cloud Computing, déjà existantes ou encore à créer.[22]
- **Fiabilité et disponibilité :** Il est important que les service cloud soient solides et fiable [23]

- **La portabilité :** la possibilité de déplacer des applications de cloud computing d'une plate-forme d'un fournisseur à une autre. Cela est difficile car les fournisseurs de services de cloud computing n'utilisent pas tous les mêmes modèles et technologies.[23]
- **Puissance de calcul :** les applications gourmandes en données nécessitent une connectivité réseau élevée, ce qui augmente les coûts. La faible capacité de transfert de données ne couvre pas les performances attendues des applications de cloud computing. [23]

I.12 Conclusion

Le Cloud Computing est considéré comme la nouvelle révolution de l'informatique à travers l'internet. Il offre des avantages pour les organisations et les individus. Dans ce chapitre, nous avons introduit plusieurs concepts du Cloud Computing. Nous avons présenté les diverses définitions données au Cloud, ses caractéristiques, ainsi que ses différents modèles de déploiement. Enfin,nous avons parlé de l'évolution du Cloud Computing qui dépend de l'évolution du multi-Cloud et de l'inter-Cloud.
Dans le chapitre suivant, nous abordons la notion de description des services cloud, les différents travaux connexes pour la description et la transformation.

CHAPITRE II

DESCRIPTION DES SERVICES CLOUD

II.1 Introduction

Il existe une variété de langages de modélisation bien conçus au service des services Internet , mais aucun d'entre eux n'est capable de décrire les fonctionnalités spéciales des services cloud , du point de vue technique , sémantique , opérationnel et commercial . C'est dans cette optique que plusieurs chercheurs se sont intéressé à la description des services cloud en utilisant différentes méthodes - les ontologies , les langages , les modèles et normes à travers les types de services Cloud (laaS , PaaS et SaaS).

Dans ce chapitre nous présentons un aperçu des éléments de base d'une description. Nous commençons par definir certains concepts propres à la description et la transformation des modèles , ensuite nous donnons une vue d'ensemble sur les langages de descriptions.

II.2 Généralités et définitions

II.2.1 Définition de la description de service

Une description de service est un critère qui permet de capturer les caractéristiques fonctionnelles et non fonctionnelles d'un service. Lors de la description d'un service, les propriétés, capacités et contraintes du service doivent toutes être prise en compte. [24]

II.2.2 Description syntaxique

- **Description structurelle :**Représente une interface sous la forme d'un standard W3C (World Wide Web Consortium) qui définit la structure abstraite et concrète d'un service (nom de l'opération, nom et type des paramètres d'entrée/sortie) sous la forme d'un document XML signé décrivant les opérations fournies par le service. [25]
- **Description comportementale :** Décrit l'interaction et l'ordre dans lequel les opérations d'un ou plusieurs services sont appelées. Un service peut avoir plusieurs descriptions de comportement, dont chacune est une vue du comportement possible du service dans une communication donnée. [25]

II.2.3 Description sémantique

La sémantique désigne les technologies visant à rendre le contenu des ressources de description « compréhensibles » (ou plus simplement interprétables) par les programmes ou agents

logiciels. [26]

• **Définition et principe du web sémantique**

Le Web sémantique est une extension du Web dont les informations sont construites selon une sémantique bien définie. Le consortium web a défini un cadre général qui permet le partage et la réutilisation des données au travers de différentes applications. Les données du web cachées dans le code HTML manquent d'information sémantique les décrivant. Seul l'homme peut lire et appréhender le contenu du site web.
Le besoin de chercher et retrouver facilement et rapidement les informations pertinentes et complètes est de plus en plus demandé par les utilisateurs du Web.
Les supports actuels construits spécifiquement pour le Web, basés sur la syntaxe des documents, ne satisfont plus ce besoin. Il faut que des agents logiciels aident plus efficacement différents types d'utilisateurs dans leur accès aux ressources sur le Web. Ainsi, le web devra devenir un espace d'échanges d'informations entre les agents humains et machines. [55]

• **Les ontologies et le web sémantique**

Une ontologie est définie comme "la science ou l'étude de l'être", qui est un ensemble de termes et de connaissances, y compris le vocabulaire, la sémantique des interconnexions, et de simples règles d'inférence et de logique pour certains domaines populaires. Les ontologies sont utilisées par les personnes, les bases de données et les applications qui doivent partager l'information du domaine (la médecine, l'imagerie, la réparation automobile, etc.). Une importante réalisation est le développement d'une nouvelle génération de langages web, qui permettent la création d'ontologies de n'importe quel domaine, et leurs instanciations dans la description de sites web spécifiques. Les ontologies sont explicitement mentionnées dans un langage formel, où plusieurs ont été proposés dans la littérature. [31]

II.2.4 Découverte de service

La découverte des services web est l'opération qui permet de « localiser une description de service web exploitable et inconnue par les machines sur un réseau, et qui répond à certains critères fonctionnels ou/et non fonctionnels » [28]
Un système de découverte devrait pouvoir identifier une similitude entre les spécifications des services utilisateurs et les offres fournis par les fournisseurs de services. Il existe deux mécanismes de découverte :

- **La découverte syntaxique :** Généralement basée sur une comparaison entre les mots-clés extraits des requêtes des utilisateurs et les mots-clés extraits des descriptions de services. Le principal inconvénient de la découverte syntaxique est qu'elle ne prend pas en compte les aspects sémantiques de la requête.
- **La découverte sémantique :** Elle consiste à utiliser différents concepts trouvés dans l'ontologie ou des ressources sémantiques externes pour calculer le degré de correspondance entre les mots-clés de requête et les mots-clés de description de service.

II.2.5 La sélection

La sélection consiste a choisir, parmi les services découverts, le service qui répond le mieux aux besoins de l'utilisateur selon des besoins fonctionnels et/ou non fonctionnels. Cette découverte est fondée sur les informations contenues dans les descriptions des services publiés. Il peut y avoir plus d'un service qui peut répondre aux besoins. Par conséquent, le meilleur service doit être choisi. [29]
Ils existent plusieurs approches de sélection de services Cloud selon le contexte de sélection de service et les méthodes utilisées, parmi ces méthodes : la méthode MCDM (Multiple Criteria Decision Analysis) et la méthode AHP(Analytic Hierarchy Process).

II.2.6 Composition de services

Plusieurs définitions ont été proposées pour le concept de composition, nous citons dans ce qui suit les plus communes : la composition est un processus qui permet de générer des services complexes à partir de services atomiques. Ces derniers ont la faculté de négocier et interagir de façon intelligente afin de découvrir automatiquement d'autres services qui servent à l'enrichissement du service composite.[27]

La composition de services web est le processus de construction de nouveaux services web à valeur ajoutée, à partir de deux ou plusieurs services web déjà présents et publiés sur le web. Un service Web est dit composé ou composite lorsque son exécution implique des interactions avec d'autres services web, des échanges de messages entre eux afin de faire appel à leurs fonctionnalités. La composition de services web spécifie quels services ont besoin d'être invoqués, dans quel ordre et comment gérer les conditions d'interaction.[73]

II.2.7 Service web

Selon W3C [1] **:** un service Web est un logiciel conçu pour soutenir l'interaction interopérable entre machines sur un réseau. Il dispose d'une interface décrite dans un format (WSDL)exploitable par des machines. D'autres systèmes interagissent avec le service Web d'une manière prescrite par sa description en utilisant des messages SOAP, qui sont généralement transmis en utilisant HTTP avec une sérialisation XML en conjonction avec d'autres normes liées au Web. [30]

Selon IBM [2] **:** Les services web sont la nouvelle vague des applications web. Ce sont des applications modulaires, auto-contenues et auto-descriptives qui peuvent être publiées, localisées et invoquées depuis le web. Les services web effectuent des actions allant de simples requêtes à des processus métiers complexes. Une fois qu'un service web est déployé, d'autres applications (y compris des services web) peuvent le découvrir et l'invoquer.

• **La communication entre les services web**

Différentes applications utilisent diverses technologies et langages de programmation. Un service Web fournit la plate-forme commune permettant à ces applications de communiquer entre elles. Par exemple, une application Java peut communiquer avec une application PHP ou .NET à l'aide de services Web via le réseau. Un service Web fournit simplement une plate-forme indépendante du langage pour assurer la communication entre diverses technologies. SOAP et REST sont deux types de services Web.

- **SOAP :** [33] est une spécification de communication entre services web, par échange de messages en XML au travers du web. Simple et facile à implémenter dans les serveurs web ou dans les serveurs d'application, SOAP est indépendant des langages de programmation ou des systèmes d'exploitation employés pour l'implémentation des services web. SOAP est un protocole de communication entre applications fondé sur XML, visant à satisfaire un double objectif : servir de protocole de communication sur les intranets, dans une optique d'intégration d'applications d'entreprise, et permettre la communication entre applications et services web, en particulier dans un contexte d'échanges inter-entreprises (le réseau internet) [32].
- **REST :** est une architecture Web standard qui réalise la communication de données à l'aide d'une interface standard telle que HTTP ou d'autres protocoles de transfert qui utilisent l'URI (Uniform Resource Identifier) standard. La conception est telle que chaque

1. *www.w3c.org*
2. *http ://www-106.ibm.com/developerworks/webservices/.*

composant d'un service Web RESTful est une ressource accessible à l'aide de méthodes HTTP standard (si le protocole choisi est HTTP). Les ressources qui peuvent être considérées comme des objets dans le concept de programmation orientée objet (POO) sont identifiées par des URI et les ressources sont représentées de plusieurs manières telles que JSON, XML, texte, etc. bien que JSON soit actuellement le choix le plus privilégié.[42]

II.3 Transformation de modèle

• **Définition de système**

Un système est une construction théorique que forme l'esprit sur un sujet (par exemple, une idée qui est mise en œuvre afin d'expliquer un phénomène physique qui peut être représenté par un modèle mathématique).

• **Définition de modèle**

Un modèle est une description abstraite d'un système construit dans un but donné. Il doit pouvoir être utilisé pour répondre à des questions sur le système. Les modèles doivent en général respecter des contraintes décrites par un méta-modèle [50].

• **Définition de méta-modèle**

Un méta-modèle est un modèle d'un langage de modélisation. Le terme "méta" signifie transcendant ou au-dessus. Un méta-modèle décrit un langage de modélisation a un niveau d'abstraction supérieur que le langage de modélisation lui-même[50].

II.3.1 Définition de transformation de modèle

Une transformation est la génération automatique d'un modèle cible à partir d'un modèle source, selon une définition de transformation. Une définition de transformation est un ensemble de règles de transformation qui décrivent comment un modèle source peut être transformé en un modèle cible [49].
Une règle de transformation est une description de la manière dont une ou plusieurs structures du langage source peuvent être transformées en une ou plusieurs structures du langage cible[49].

II.3.2 Les modèles de transformation

Une transformation de modèle est le processus de conversion d'un modèle source en un modèle cible selon un ensemble de règles de transformation. Les règles sont définies avec un langage de transformation de modèle. Une règle de transformation se compose de deux parties : une partie gauche qui accède au modèle source ; et une partie droite qui se développe sur le modèle cible. Le modèle source est conforme au méta-modèle source et le modèle cible est conforme au méta-modèle cible. Lors de la transformation, le modèle source reste inchangé.
Si les méta-modèles source et cible sont identiques, la transformation est dite endogène, sinon elle est qualifiée d'exogène. Si le niveau d'abstraction ne change pas, la transformation est appelée transformation horizontale. Si le niveau d'abstraction change, la transformation est appelée transformation verticale.
Selon le type et la complexité du processus, la transformation du modèle peut être nommée **transformation modèle vers modèle (M2M)**, **transformation modèle vers texte (M2T)** [51].

- **Transformation modèle vers modèle (M2M)**

  La transformation de modèle à modèle (également connue sous le nom de M2M ou MMT) est un projet qui héberge des langages de transformation de modèle à modèle. Avec ces langages M2M, nous pouvons définir des transformations de modèles pour produire d'autres modèles. En général, le mécanisme du langage M2M consiste à prendre un modèle (conforme à un méta-modèle source) en entrée, à spécifier des règles de transformation et générer un modèle cible (conforme à un méta-modèle cible)[51].

- **Transformation modèle vers texte (M2T)**

  La transformation de modèle à texte (également connue sous le nom de M2T)est un projet qui héberge des langages de transformation de modèle à texte. La transformation M2T convertit un modèle source en un fichier texte. La transformation M2T est généralement utilisée pour la génération de code final ou la création de documents de support [51] .

II.3.3 Langage de transformation

Plusieurs langages de transformation ont été proposés pour réaliser des transformations de modèle selon l'approche par méta-modélisation. Parmi ces langage, le langage ATL.

- **Atlas Transformation Language "ATL"**

  Est un langage et une boîte à outils de transformation de modèles développés par le groupe Atlas (INRIA et LINA).[44] C'est un langage de transformation de modèle spécifié à la fois comme métamodèle et comme syntaxe textuelle concrète. Un ATL programme

de transformation est composé de règles qui définissent comment les éléments du modèle source sont mis en correspondance pour créer et initialiser les éléments des modèles cibles.[45]

II.4 Les langages de description de service

II.4.1 XML (eXtensible Markup Language)

XML est un standard du W3C(l'organisme chargé de standardiser les évolutions du web). On retrouve dans XML une généralisation des idées contenues dans HTML et SGML. XML permet de définir des balises extensible et indépendante de l'aspect graphique que doit revêtir la page dans le navigateur web. Dans XML, les balises permettent d'associer toutes sortes d'informations au fil du texte.[32]
Il a été conçu pour des documents arbitrairement complexes, tout en s'appuyant sur cinq grands principes simples et clairs [33] :

- Lisibilité à la fois par les machines et par les utilisateurs.
- Définition sans ambiguïté du contenu d'un document.
- Définition sans ambiguïté de la structure d'un document.
- Séparation entre documents et relation entre documents.
- Séparation entre structure des documents et présentation des documents.

II.4.2 WSDL (Web Services Description Language)

Le standard WSDL est un langage basé sur XML permettant de décrire les services web, spécifier leurs localisations et les opérations qu'ils exposent [34]. WSDL représente les services web sous forme de points de communications ou de ports échangeant des messages [35]. Ces ports sont mis bout à bout et reliés par un protocole de transport. WSDL permet de séparer clairement entre la définition abstraite et les mécanismes de liaison (protocoles de transport et format de données). Cela permet de réutiliser les définitions abstraites : messages et portTypes. Les formats de données et les protocoles de transport concrets d'un portType constituent un binding réutilisable. SOAP est généralement utilisé comme protocole d'accès, mais on peut aussi utiliser HTTP, ou tout autre protocole. Un port est défini en associant une adresse réseau à un binding. Un ensemble de ports définit un service [34][35]. De façon plus précise, Un document WSDL utilise les éléments suivants pour décrire un service web [34] :

1. **Types :** les types de données que l'on peut employer dans les messages, en référençant

en général les types standards des schémas XML.

2. **Message :** les messages sont des descriptions abstraites des données échangées.
3. **Operation :** les opérations sont des actions élémentaires que l'on peut effectuer auprès du service.
4. **PortType :** définit un ensemble d'opérations abstraites.
5. **Binding :** spécifie un protocole et un format de données concret pour un portType.
6. **Port :** les ports sont des points de communication constitués d'un binding et d'une adresse réseau.
7. **Service :** les services sont des collections de ports.

II.4.3 OWL-S (Web Ontology Language for Web Services)

OWL-S est une ontologie de description des services web, dont les objectifs sont de résoudre les ambiguïtés et de rendre la description d'un service compréhensible par une machine [37].Dans [38] les auteurs présentent une ontologie pour les services web dans le but d'automatiser la découverte, l'invocation, la composition et la surveillance de l'exécution des services. Les auteurs reprennent la notion de classes d'OWL et proposent l'ontologie OWL-S. OWL-S fournit trois connaissances essentielles sur les services Web, qui répondent aux questions suivantes :

• que fournit le service Web pour l'utilisateur potentiel ? La réponse à cette question est fournie par le ServiceProfile,

• comment fonctionne le service Web ? La réponse à cette question est fournie par le ServiceModel,

• comment le client peut-il interagir avec le service Web ? La réponse à cette question est fournie par le ServiceGrounding. OWL-S décrit donc les services web suivant trois points de vue différents qui sont le ServiceProfile, le ServiceModel et le ServiceGrounding [37].

• **ServiceProfile**, qui indique "ce que fait le service".

• **ServiceModel**, qui indique "comment le service fonctionne".

• **ServiceGrounding**, qui indique "comment accéder au service".

II.4.4 USDL (Unified Service Description Language)

USDL est un langage permettant de décrire les services Internet sous trois angles différents (technique, commercial et opérationnel). Les aspects techniques sont couverts par les modules

Technologie et Fondamentaux de l'USDL Les aspects opérationnels sont couverts par les modules Services, Fonctions et Fondamentaux de l'USDL. Les aspects commerciaux sont décrits par les modules Participants, Interactions, Tarification, Juridique, Niveaux de service et Fondamentaux de l'USDL [47]

II.4.5 WSMO (Web Service Modeling Ontology)

WSMO est un méta-modèle pour les aspects liés aux services Web sémantiques. La spécification MOF (Meta-Object Facility) est utilisée pour spécifier ce modèle. MOF définit un langage abstrait et un cadre pour spécifier, construire et gérer des méta-modèles technologiquement neutres. [53]

WSMO identifie quatre éléments de niveau supérieur comme concepts principaux qui doivent être décrits afin de décrire les services Web sémantiques.[53]

- **Les ontologies :** fournissent la terminologie utilisée par d'autres éléments WSMO pour décrire les aspects pertinents des domaines.
- **Les services Web :** décrivent l'entité de calcul fournissant l'accès aux services qui fournissent une certaine valeur dans un domaine. Ces descriptions comprennent les capacités, les interfaces et le fonctionnement interne du service Web . Tous ces aspects d'un service Web sont décrits en utilisant la terminologie définie par les ontologies.
- **Goals :** représentent les désirs des utilisateurs, dont la réalisation pourrait être recherchée en exécutant un service Web. Les objectifs modélisent la vue de l'utilisateur dans le processus d'utilisation du service Web.
- **Les médiateurs :** décrivent des éléments qui surmontent les problèmes d'interopérabilité entre différents éléments WSMO. Les médiateurs sont le concept de base pour résoudre les incompatibilités au niveau des données, des processus et des protocoles.

II.4.6 Comparaison entre les langages de description de service

Le tableau II.1 illustre une comparaison entre les langages de description de service suivant les critères suivants : domaine, représentation, aspects couverts, les plateformes couvertes et langage de base.

Langage	Domain	Représentation	Aspect	Plateforme	Langage de base
XML	General	Ensemble des balises	Selon les informations	Multiplateforme	Des balises
USDL	General	XML	Commerciaux Opérationnelle Technique	Multiplateforme	MOF-basé Sur Meta-modèle
OWL-S	Sémantique web service	XML	Technique Opérationnelle	Multiplateforme	OWL/RDF
WSDL	Sémantique web service	XML	Technique	Multiplateforme	XML
WSMO	Sémantique web service	XML	Technique	Multiplateforme	OWL/RDF

TABLE II.1 – Comparaison entre les langages de description de service

II.5 Conclusion

Dans ce chapitre, nous avons donné un aperçu des éléments de base d'une description, suivi d'une étude de transformation de modèle avec les types existant et les langages utilisés pour la transformation. Après, nous avons fait une étude comparative entre les langages de descriptions.

Dans le chapitre suivant, Nous avons présenté certains travaux connexes sur les approches de description des services cloud et les approches de transformation pour savoir ce qui existait déjà dans la littérature et ce qui manquait afin de donner une meilleure proposition..

CHAPITRE III

SOLUTIONS EXISTANTES POUR LA DESCRIPTION DES SERVICES DANS LE CLOUD

III.1 Introduction

Les problèmes mentionnés dans le chapitre précédent ont fait l'objet de plusieurs efforts ayant visé la résolution de la problématique de description. Dans ce chapitre nous présenterons ces efforts. Nous discuterons les solutions proposées pour réaliser la description dans le Cloud Computing.

III.2 Travaux connexes

Plusieurs chercheurs se sont intéressés au la problème de l'interopérabilité des services cloud en utilisant différentes approches . Certains se sont basés sur la description de service cloud, d'autres sur la transformation des langages de description vers une description unique. Nous présentons, dans ce qui suit, quelques travaux de recherche qui traitent la description et la transformation de description pour les services cloud.

III.2.1 Approches de description

- **Travaux de [56]** ont proposé une description à base d'USDL . Ils ont représenté la description du cloud sémantiquement par une ontologie WSMO et ont fait le mapping entre USDL et WSMO .

  La solution proposée traite tous les aspects (technique , opérationnel , commercial , sémantique) et tous les types de services (IaaS , SaaS , PaaS) . Cependant, les auteurs n'ont pas traité les problèmes de sécurité et ils n'ont pas défini de SLA entre les différentes couches .
- **Travaux de [57]** mettent en place le projet Aggregating laaS service qui permet de rassembler des services laaS de différents fournisseurs en utilisant une même interface commune. Ils proposent modèles pour décrire le calcul , le stockage , le réseau et la machine virtuelle.
- **Travaux de [58]** ont défini un méta - modèle pour une application cloud qui est capable de capturer la syntaxe et la sémantique de certaines des applications cloud . Le méta - modèle proposé montre les principaux éléments de vocabulaire et de conception cloud , l'ensemble des règles de configuration et l'interprétation sémantique . La solution proposée couvre les aspects opérationnels , technique , sémantique et néglige l'aspect commercial , de plus elle concerne seulement le SaaS.
- **Travaux de [59]** proposent un langage de spécification pour modéliser l'organisation du cloud de l'infrastructure en tant que service . Ils ont décrit une application cloud en

construisant une architecture : architecture du cloud trust , dans laquelle les modèles cloud sont combinés avec attestation à distance pour fournir des services de confiance .Cependant , cette solution permet seulement la description de type IaaS et les auteurs ne se focalisent que sur l'aspect technique et négligent les aspects commercial , sémantique et opérationnel .

- **Travaux de [60]** identifient les informations nécessaires aux organisations pour la sélection des services cloud en utilisant le langage de description de service unifié (USDL) comme solution . Ils affinent alors chaque module vers le domaine des services de cloud computing en intégrant les caractéristiques de service de cloud requis par les clients . Cela aide les clients à structurer leurs sélections de services ainsi que les fournisseurs de cloud dans la création de descriptions de services en fonction des besoins du client . La solution proposée concerne tous les types de service cloud à savoir laaS, PaaS , SaaS et couvre l'aspect technique , opérationnel et commercial.
- **Travaux de [61]** proposent un modèle de description de service cloud sémantique unifié , CSDM . Le CSDM proposé a été étendu à partir de la structure de base de l'USDL en définissant un module supplémentaire , le module de transaction , qui modélise le système de notation des services de cloud à partir de plusieurs aspects , tels que le risque , la confiance et la réputation . Les chercheurs ont également conçu un système d'annotation basé sur OWL pour enrichir l'expressivité sémantique de ce modèle. La solution proposée traite tous les aspects (technique , opérationnel , commercial , sémantique) et tous les types de services (IaaS , SaaS , PaaS).
- **Travaux de [62]** La solution (Blueprinting) proposée dans [62] est le résultat d'un ensemble de tests sur certaines lacunes des offres de services actuelles et certaines approches de gestion afin de trouver une solution globale sur la base de concept et des techniques de cloud Blueprinting. Le concept Blueprint est une description abstraite et uniforme des offres de services cloud inter-couches , un modèle Blueprint pour décrire les plans , et un cycle de vie Blueprint qui explique comment les plans sont utilisés pendant toutes les phases d'ingénierie d'un SBA (Service - based Application). La solution Blueprinting concerne toutes les couches du cloud computing à savoir laaS , PaaS , SaaS et couvre les aspects technique et commercial.
- **Travaux de [63]** Les auteurs ont entrepris une étude préliminaire sur la découverte de services cloud en adoptant un paradigme P2P non structuré . Ils ont mis au point un mécanisme pour le routage des demandes de service en couplant un certain nombre de composants : la réplication d'un saut , sémantique , réorganisation de la topologie et des

noeuds. Ils ont commencé en utilisant WSDL - S , dans WSDL - S , l'expressivité du WSDL a été augmentée avec la sémantique en adoptant des concepts similaires à ceux de OWL - S, ils ont ensuite progressivement amélioré WSDL - S au moyen de l'extensibilité fournie par WSDL lui - même selon les exigences spécifiques pour les descriptions de services Cloud. La solution couvre les aspects technique et sémantique au niveau du type de service SaaS.

- **Travaux de [64],[65]** ont développé un ensemble d'ontologies OWL pour saisir et exprimer les aspects commerciaux des offres des fournisseurs et des demandes des clients . Le but de leur travail est de soutenir un processus sémantique avancé d'appariement entre l'offre et la demande et de proposer un modèle sémantique pour décrire les caractéristiques liées aux affaires des ressources du cloud . Une procédure de jumelage a également été conçue pour rechercher sémantiquement le domaine des offres et fournir au client une liste des offres les plus rentables . Cependant , la solution proposée néglige l'aspect technique et ne concerne que le SaaS et le PaaS.
- **Travaux de [66]** ont proposé une architecture de référence comme modèle de fonctionnalités pour décrire les offres de services de manière standardisée. Ce modèle de fonctionnalités comprend neuf fonctionnalités telles que : la description du type de service, déploiement, tarification, rôle, aspects d'intégration de services, options d'approvisionnement, SLA, aspects spécifiques à l'organisation, et les caractéristiques du service cloud. Ils ont, également, développé une technique de visualisation appelée Cloud service navigateurs (CSN) pour décrire visuellement les offres et demandes de services cloud et les comparer. Cette technique permet de détecter rapidement les descriptions de service.La solution proposée se concentre sur l'aspect technique.
- **Travaux de [67]** ont proposé un modèle de représentation pour stocker le service d'infrastructure en langage XML. Ce modèle décrit différentes caractéristiques de service cloud. Il comprend le nom du service, les propriétés fonctionnelles et non fonctionnelles et l'emplacement du serveur. Ces propriétés peuvent être qualitatives ou quantitatives. Le modèle de représentation des données proposé est utilisé par les fournisseurs pour publier leurs services et pour aider les utilisateurs à découvrir les services en utilisant à la fois des informations syntaxiques et sémantiques. Ce travail ne traite pas uniquement les services d'infrastructure, mais il est destiné à couvrir tous les types de services.
- **Travaux de [68]** Les auteurs dans [68] ont adopté une approche basée sur un modèle pour définir et gérer les services cloud. Ils ont proposé un méta-modèle de service cloud nommé CoPS qui décrit le service cloud comme une composition d'éléments logiciels et matériels

à l'aide de trois modèles : composant, produit et service. Le méta-modèle CoPS offre une description uniforme du service cloud. Ce modèle est utilisé par une architecture de Cloud Management System (CMS), qui est capable de transformer dynamiquement le modèle de service d'une représentation abstraite en un déploiement réel.

- **Travaux de [69]** Les auteurs dans [69] ont proposé un modèle sémantique de description de service cloud appelé CSDM (Cloud Service Description Model), qui est une extension de l'USDL. Il couvre les aspects techniques, aspects opérationnels, commerciaux et sémantiques. Il divise les informations de service en dix modules (Service, fonctionnel, technique, interaction, participant, niveau de service, tarification, juridique, fondement et transaction) [69] a ajouté, par rapport à l'USDL de base, un module supplémentaire nommé module de transaction. CSDM permet la description des différents services avec différents modèles de livraison et de déploiement.

Discussion sur les approches de description

Après une étude des travaux de recherche sur la description des services cloud , nous résumons ces différents travaux dans le tableau II.2.

Travaux de recherche	Technique utilisée	Représentation proposée	Modèle de prestation	Aspect couvert
[56]	Ontologie	WSMO USDL	IaaS PaaS SaaS	Technique Opérationnel Sémantique Commercial
[57]	Modèle	Meta-modèle	IaaS	Technique Opérationnel
[58]	Modèle	Meta-modèle	SaaS	Technique Opérationnel Sémantique
[59]	Langage	Cloud	IaaS	Technique
[60]	Langage	USDL	IaaS PaaS SaaS	Technique Opérationnel Commercial
[61]	Modèle et langage	CSDM	IaaS PaaS SaaS	Technique Opérationnel Sémantique Commercial
[62]	Modèle	Modèle de plan directeur	IaaS PaaS SaaS	Technique Commercial
[63]	Langage	P2P	SaaS	Technique Sémantique
[64]	Ontologie OWL-S	Ontologie d'affaire	PaaS SaaS	Commercial Sémantique
[65]	Ontologie OWL-S	Ontologie d'affaire	PaaS SaaS	Commercial Sémantique
[66]	Modèle	CSN	IaaS PaaS SaaS	technique
[67]	Modèle	langage XML	IaaS PaaS SaaS	technique Opérationnel
[68]	Modèle	CoPS CMS Par UML	IaaS PaaS SaaS	Commercial technique Opérationnel
[69]	Modèle	CSDM Par USDL	IaaS PaaS SaaS	Commercial technique Opérationnel Sémantique

TABLE III.1 – Tableau comparatif des approches de description

D'après le tableau II.2 , nous pouvons remarquer que :

- 10 des travaux étudiés concernent les services de type IaaS.

- 10 des travaux étudiés concernent les services de type PaaS.
- 12 des travaux étudié les services de type SaaS.

Nous remarquons que les trois types de livraison de service (IaaS, PaaS, SaaS) ont fait l'objet de recherche pour la problématique de la description. Aussi, les techniques les plus utilisées demeurent les modèles.

III.2.2 Approches de transformation

- **Travaux de [43]** cette solution propose une approche d'interopérabilité de service cloud basée sur la standardisation des descriptions de services cloud.
  Le service appelé CIPiMo , CIPiMo est un service MaaS (Model as a Service) basé sur MDE, qui permet l'interopérabilité des services cloud en transformant des descriptions de services hétérogènes en un GCSD.

  MaaS est une variante SaaS, qui permet aux utilisateurs de déployer et d'exécuter des services de modélisation sur Internet. Il fournit une interface permettant au client de communiquer avec le prestataire de service. Parmi les principaux apports du MaaS, le fait qu'il permet l'interopérabilité entre les outils et les systèmes en comblant les écarts entre leurs spécifications.
- **Travaux de [46]** cette solution propose aussi un modèle d'interopérabilité des services basé sur un modèle CSD générique(GCSD). Le médiateur de transformation proposé utilise les règles de mapping pour transformer des descriptions de services cloud hétérogènes en la description uniforme GCSD. Par conséquent, cela les rend interopérable. Ce travail couvre seulement la transformation de OWL-S vers GCSD. Le GCSD est basé sur USDL.
  Le processus de médiation repose sur un ensemble de règles de mapping proposées pour OWL-S avec utilisation de modèle pivot pour la transformation. Les règles proposées mappent les concepts et les propriétés à partir d'un langage source CDS à leurs concepts équivalents dans le GCSD. Par conséquent, différents services seront décrits de manière uniforme ce qui permet leur interopérabilité. Le modèle proposé est une solution d'interopérabilité centrée sur le client et sur le fournisseur sur différents modèles de cloud (SaaS, PaaS, IaaS).

• **Discussion sur les approches de transformation**

Après une étude des travaux de recherche sur la transformation de description des services cloud, nous les résumons selon les critères suivants : le principe, le modèle de présentation, la transformation , le type de description , le type de transformation , le langage de GCSD et les aspects couverts (Tableau II.3) .

Solution	Principe	Modèle de présentation	Transformation	Type de description	Type de transformation	Langage de GCSD	Aspect couvert
[43]	CSD vers GCSD	PaaS IaaS SaaS	M2M transformation Langage ATL	WSDL OWL-S	Pivot	USDL Basé sur WSMO ontologie	Technique Opération
[46]	CSD vers GCSD	PaaS IaaS SaaS	Ensemble de règle	OWL-S	Pivot	USDL Basé sur WSMO ontologie	Technique Opération

TABLE III.2 – Tableau récapitulatif des travaux de transformation

D'après la comparaison des travaux, nous pouvons remarquer que :

- Les solutions donnent une description générale de Cloud basée sur USDL et transforment l'autre langage en cette description.
- Les solutions se basent sur les 3 types de service PaaS, IaaS et SaaS.
- Les chercheurs donnent un intérêt à la description OWL-S.
- Les transformations utilisées sont basées sur la méthode pivot.
- Les solutions ne traitent pas tous les aspects de la description.

• **Discussion de la solution [43]**

Cette solution est basée sur la transformation de CSD vers GCSD en utilisant les règles de transformation avec le langage ATL. Cette méthode oblige les fournisseurs à transformer leurs descriptions en GCSD et à travailler avec cette description. Pour cela, les fournisseurs qui utilisent cette méthode sont verrouillés avec la description GCSD. En plus, pour transformer en GCSD il faut que la description de service soit une description WSDL et OWL-S. La description GCSD est écrite en langage USDL et se base sur WSMO pour répondre aux 4 aspects (technique, commerciaux, opérationnel et sémantique) et ajoute les notions de Cloud computing.Cependant, le WSDL est basé sur l'aspect technique seulement et les auteurs n'essayent pas d'enrichir le

wsdl pour traiter les 4 aspects et les notions de Cloud. Aussi, le OWL-S traite seulement l'aspect opérationnel et technique sans les notions de Cloud.

•Discussion de la solution [46]

Cette solution est basée sur la transformation de CSD vers GCSD en utilisant un ensemble de règles de transformation avec un pivot. Cette solution cause aussi le verrouillage avec la description GCSD , le fournisseur oblige à travailler avec la description de GCSD pour éviter l'interopérabilité entre les services. Les auteurs donnent des règles de transformation pour le langage OWL-S seulement , donc il faut définir le service avec la description OWL-S pour la transformer vers GCSD.

III.3 Discussion

Après, une étude des travaux de recherche sur la description des services cloud et les approches de transformation, nous remarquons que la plupart des chercheurs ont travaillé sur les approches de la description qui proposent une description générale pour le cloud et la majorité des études se base sur le modèle de prestation SaaS. Aussi, la technique la plus utilisée est la technique modèle.
Concernant les approches de transformation, elles consistent à donner des règles pour transformer les langages des descriptions qui sont,en général, owl - s et wsdl en description cloud générale.

III.4 Conclusion

Dans ce chapitre nous avons présenté des travaux proposés dans le contexte de la description de service et la transformation pour avoir une idée sur les différentes solutions proposées. Nous avons terminé le chapitre par une étude comparative des différentes approches proposées, tenant compte de plusieurs critères, pour voir la différence entre ces approches et pour définir les objectifs de recherche pour chaque approche.

Dans le chapitre suivant, nous allons expliquer notre solution qui comporte trois étapes de base : la description de service, adaptation des langages au cloud et enfin la transformation des langages vers le langage général.

CHAPITRE IV

CONCEPTION

IV.1 Introduction

Après avoir étudier les solutions existantes pour la description des services Cloud dans le chapitre précédent , nous allons dans ce chapitre ,présenter notre solution , qui est une description générique de service cloud gérant l'interoperabilité au niveau technique, opérationnel, sémantique et commercial dans un environnement de services de Cloud Computing de type SaaS,PaaS et IaaS .
L'objectif de ce chapitre est de prouver que nous pouvons décrire tous les aspects d'un service cloud,. Pour cela, nous allons expliquer l'adaptation des différentes descriptions au cloud computing et le mapping des langages de description vers la description générale proposée.

IV.2 Démarche de travail

La démarche de travail que nous avons suivi afin de réaliser notre projet est l'XP-eXtreme Programming [1] . L'eXtreme Programming (XP) est une méthode agile [2] de gestion de projet informatique adaptée aux équipes réduites avec des besoins changeants.
L'Extreme Programming repose largement sur le travail d'équipe car tous les membres de l'équipe sont considérés comme des participants égaux qui travaillent de manière très productive et efficace.
L'Extreme Programming applique cinq principes lorsqu'il s'agit de travailler sur un projet :

- **Communication :** il est essentiel que chaque membre de l'équipe communique quotidiennement avec ses collègues ainsi qu'avec le client. C'est un moyen incontournable pour résoudre les problèmes.
- **Simplicité :** la façon la plus simple d'arriver au résultat est privilégiée. L'équipe projet fait ce qui est nécessaire et demandé, rien de plus. Une application simple sera plus facile à faire évoluer ensuite.
- **Feedback :** le retour d'information entre l'équipe projet et le client est essentiel. Chaque étape du projet est envoyée aussi rapidement et souvent que possible au client afin qu'il teste, donne son avis et valide l'étape. Chaque demande de modification est prise en compte immédiatement.
- **Respect :** le respect de chaque membre de l'équipe et de son travail sont primordiaux.
- **Courage :** Il faut du courage pour effectuer certains changements comme essayer une

1. http ://www.extremeprogramming.org/
2. Une méthode agile est une approche itérative et collaborative, capable de prendre en compte les besoins initiaux du client et ceux liés évolutions. Les méthodes méthodes agiles sont basées sur un cycle de développement centré sur le client. Les clients interviennent dans la réalisation du début à la fin du projet.

nouvelle technique, recommencer une itération non validée ou revoir l'organisation du projet. Le courage permet de sortir d'une situation inadaptée.

IV.3 La description de cloud proposée

Notre proposition consiste à donner un modèle UML regroupant toutes les notions de cloud computing computing, en se basant sur les 4 aspects (technique, opérationnel, sémantique et commercial), pour les 3 modèles de prestation (IaaS, PaaS et SaaS).
Dans ce qui suit, nous présentons les exigences nécessaires pour une meilleure description de service cloud.

Différents modèles de prestation de services :

La prestation de service cloud computing est une notion importante qu'il faut prendre en charge dans toute description générique du cloud computing.La figure III.1 représente les 3 types de prestation dans notre modèle.

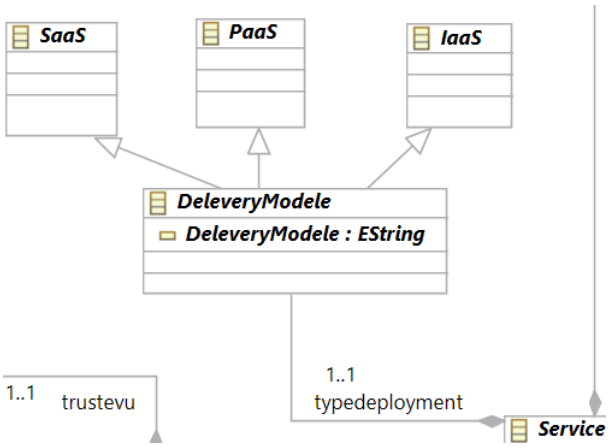

FIGURE IV.1 – Différents modèles de prestation de services

Différents modèles de déploiement de services :

Les services cloud sont déployés selon différents modèles de déploiement (public, privé, communautaire et hybride). Ainsi, le modèle de description de service doit décrire les connaissances sur ces modèles de déploiement. La figure III.2 représente les 4 modèles de déploiement de services.

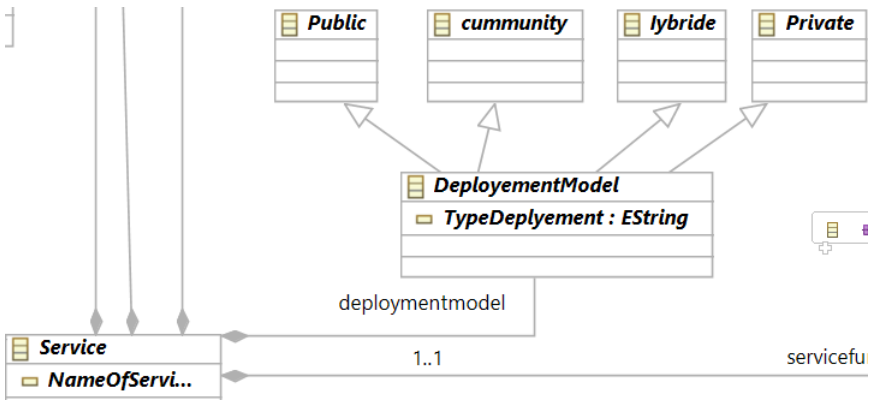

FIGURE IV.2 – Différents modèles de déploiement de services

Mesures des attributs de service :

Les mécanismes d'évaluation sont très importants dans l'environnement Cloud Computing, car ils aident le consommateur cloud, et plus précisément, le non-technicien pour choisir et sélectionner les services. De plus, les évaluations permettent au fournisseur de services de comparer ses services à d'autres disponibles sur le marché.
Des mécanismes de mesure sont également utilisés pour détecter les violations SLA.
Ce module capture les concepts utilisés pour mesurer et évaluer les facteurs pertinents pour la transaction de service. La figure III.3 représente les Mesure des attributs de service.

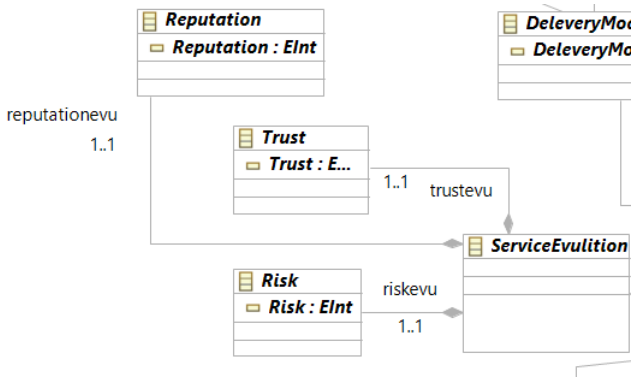

FIGURE IV.3 – Mesures des attributs de service

Description des différents acteurs :

Le Cloud Computing comprend cinq acteurs : le consommateur, le fournisseur, le courtier, l'auditeur et le transporteur. Ainsi, la description du service Cloud doit décrire ces différents

acteurs, leurs relations et leurs interactions avec leurs prestations. Dans notre modèle, nous donnons plus de détails. Nous avons classé les agents selon l'organisation et le rôle de chaque personne «voir la figure III.4 ». Les personnes sont classés en 2, les clients et les services agent. Les clients regroupent les « ConsummerSkills et ConsummerPreferences ». Le service agent regroupe les notions suivantes « Provider, Creator, Publisher, Contributor, Broker, Carrier et Auditor ». « Voir la figure III.5 ».

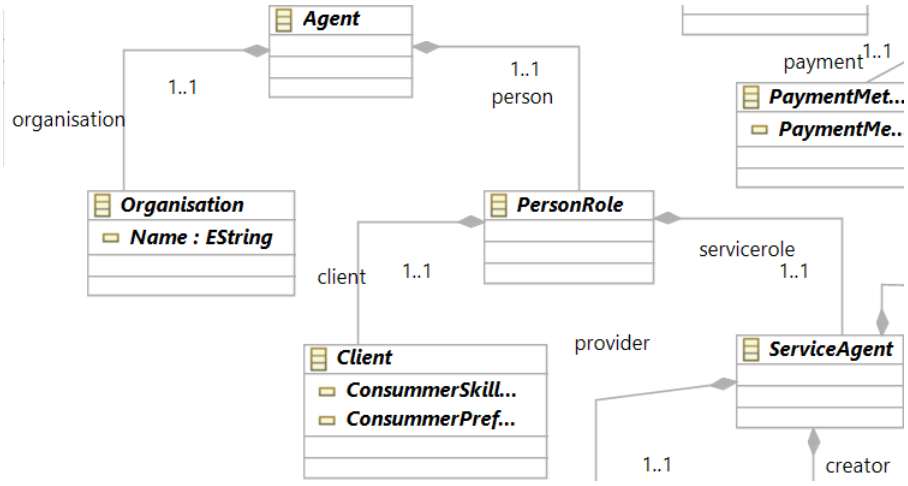

FIGURE IV.4 – Les clients de service cloud

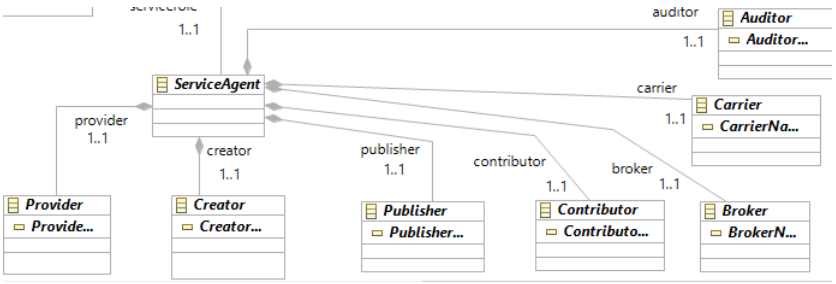

FIGURE IV.5 – Les services agent de service cloud

SLA pour les types de Cloud :

Après avoir sélectionné les prestations, il est nécessaire de définir les différents éléments du contrat. Les clauses de pénalité SLA sont déclenchées et exécutées en cas de violation. Le modèle de description de service doit :

- **Bien définir le contrat SLA.**

- **Bien décrire les sanctions.**
- **Bien définir un contrat SLA entre différentes couches Cloud (SaaS, PaaS et IaaS).**

La bonne définition du SLA permet d'établir l'interopérabilité entre les différents services des trois couches, et par conséquent, l'interopérabilité entre les fournisseurs de cloud, ce qui permet d'éviter le problème de verrouillage du fournisseur. La figure III.6 représente le SLA pour les 3 types.

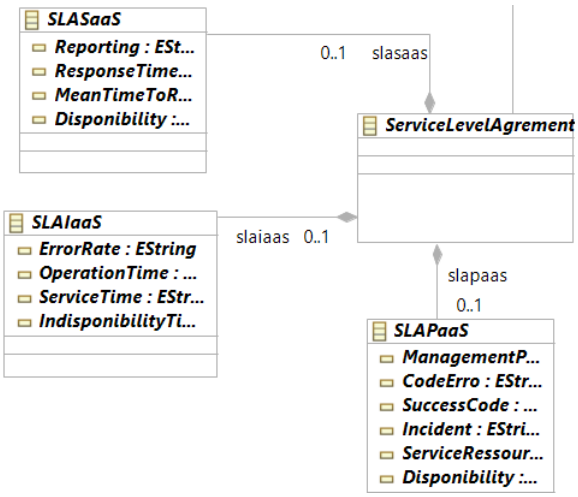

FIGURE IV.6 – SLA pour les types de Cloud

Mesure de prix : (Pricing)

Cette mesure concerne la charge des services mutuellement compris par ceux qui possèdent ou offrent des services et ceux qui les consomment. Dans notre proposition, nous avons ajouté les notions suivantes «Prix, Ajustement de prix, Mode de paiement, Clôture des prix, Mesure du prix, Impôt et Quantité » « voir la figure III.7 ».

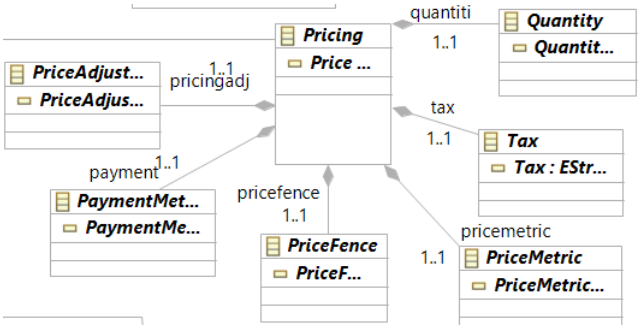

FIGURE IV.7 – Mesure de prix de cloud computing

Mesure juridique : (Legal)

Elle contient les informations sur les conditions , les licences et les droits des utilisateurs. Chaque service doit faire l'objet d'une licence. La licence inclut le droit d'utilisation. Ce dernier met d'une manière bien définie la façon d'utiliser un service (comme le droit de distribuer). Toutes ces propriétés aident l'utilisateur à sélectionner facilement les services appropriés. Notre proposition inclut dans la partie juridique les notions suivantes « Licence, Utiliser Terme, Signal point de contact, Compensation, Périmètre de responsabilité et Peine ». « voir la figure III.8 »

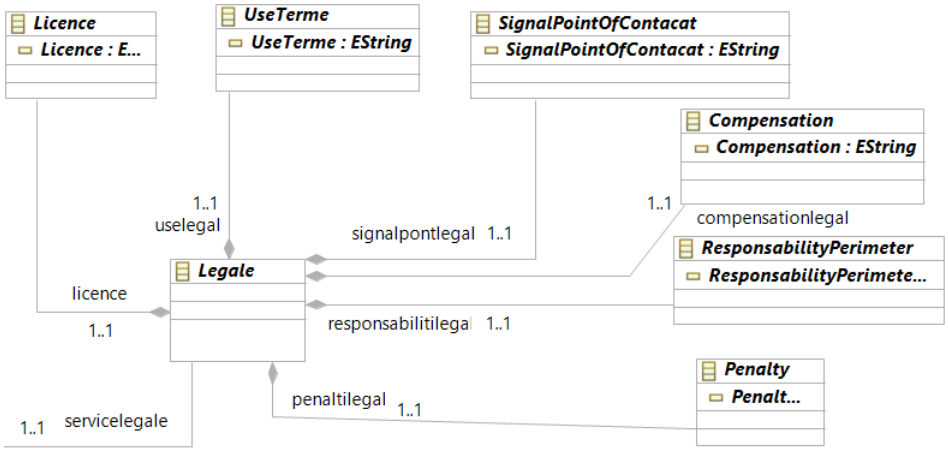

FIGURE IV.8 – Mesure juridique de cloud computing

Mesure Fondation : (Foundation)

Cette mesure fournit un ensemble de propriétés communes telles que le temps, le lieu, etc. Ces propriétés dans notre proposition contiennent 3 notions qui sont « La description, Expression et Certificat ». « Voir la figure III.9 »

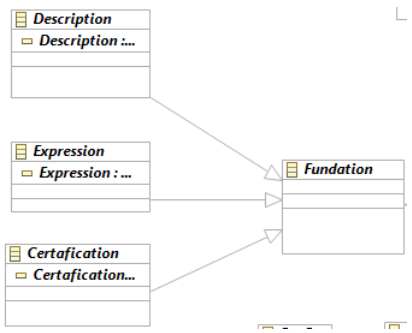

FIGURE IV.9 – Mesure Fondation de cloud computing

Mesure fonctionnel : (functional)

Ce module contient les informations relatives aux capacités de service, les paramètres d'entrée - sortie et les contraintes de chaque fonction. Nous avons ajouté la notion Interaction qui veut dire l'aspect comportemental du service. Il décrit les acteurs et les interactions entre eux. « Voir la figure III.10 ».

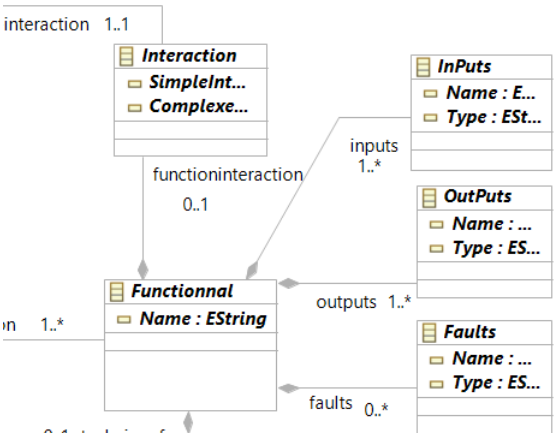

FIGURE IV.10 – Mesure fonctionnel de cloud computing

Mesure technique :

La mesure technique sert à spécifier l'association sémantique entre les descriptions techniques d'interface et les éléments de la description. Dans notre proposition la notion technique ajoute 2 éléments qui sont le protocole qui exprime le protocole de transport et URI qui exprime la localisation de service. « Voir la figure III.11 »

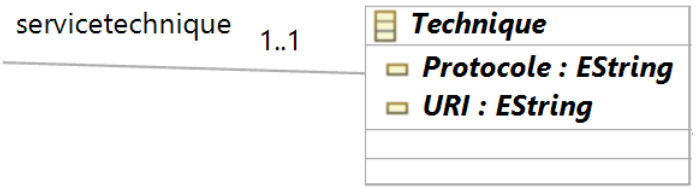

FIGURE IV.11 – Mesure technique de cloud computing

Mesure Contexte :

La mesure contexte regroupe les notions suivantes :

- **Disponibilité :** le service existe dans un environnement dynamique qui peut être modifié en raison de divers facteurs tels que : la panne du service, dégradation de la qualité du service, modification des besoins des utilisateurs ou indisponibilité des services en raison d'un problème réseau. Pour cela, nous avons ajouté la notion de disponibilité.
- **Localisation :**le service cloud existe dans un environnement variable. Son utilisation est limitée par plusieurs contraintes environnementales, qui peuvent être des contraintes d'espace. Par exemple, un consommateur souhaite utiliser uniquement les services Cloud existants en Algérie. Donc, il devrait mettre la contrainte sur l'emplacement.
- **Temps :** le service cloud existe dans un environnement variable. Son utilisation est limitée par plusieurs contraintes environnementales, qui peuvent être des contraintes de temps.
- **Ressource :** les descriptions incluent des URI qui sont nécessaires comme owl-s contient le URI d'ontologie et wsdl des xml-shema.
- **Estimation du consommateur :** les estimations faites par le client sont très importantes dans l'environnement Cloud Computing pour la mise à jours des services.

« Voir la figure III.12 »

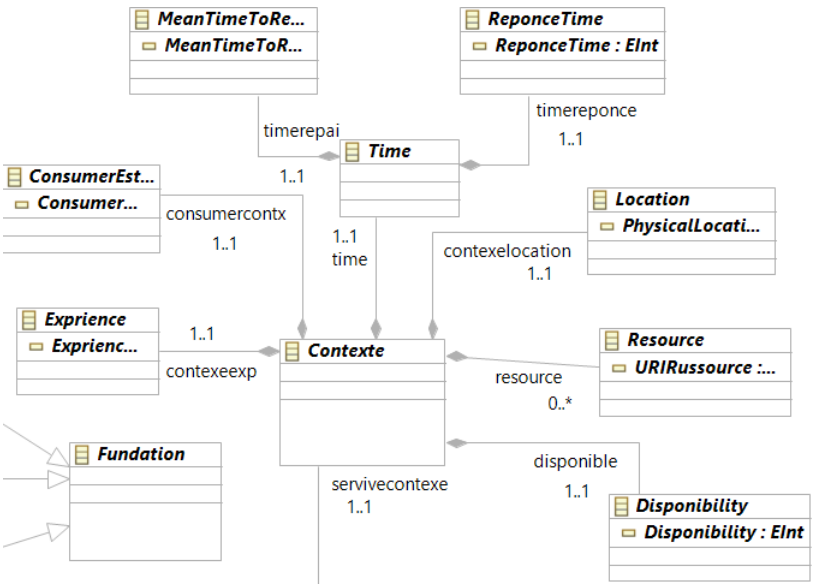

FIGURE IV.12 – Mesure Contexte de cloud computing

Service :

La racine, cette notion regroupe toutes les autres notions. Elle inclut le nom de service. « Voir la figure III.13 ».

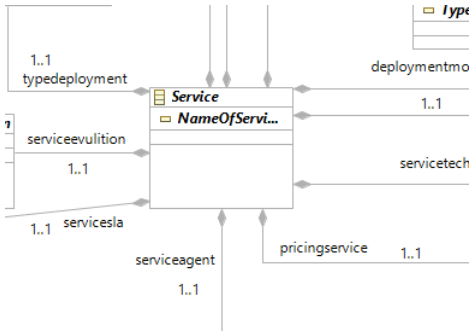

FIGURE IV.13 – La racine de description de cloud computing

IV.4 Adaptation des langages de description au cloud

Plusieurs approches ont été proposées pour décrire les services de manière riche et générique (comme WSDL, OWL-S, WSMO et USDL). Cependant, les approches actuelles restent inappropriées pour le cloud computing car :

- Elles manquent d'une manière ou d'une autre de l'aspect sémantique ou métier,
- Elles ne peuvent pas entièrement gérer les propriétés non fonctionnelles,
- Elles ne sont pas en mesure de couvrir tous les types de services (tels que SaaS, PaaS, IaaS).

Comme deuxième étape de notre proposition, nous avons décidé d'aligner les langages de description les plus populaires sur le langage de description que nous avons proposé précédemment pour couvrir tous les aspects du cloud.
Notre travail consiste à convertir les langages suivants :**WSDL, WSMO, USDL** et **OWL-S.**

IV.4.1 Adaptation de WSDL au cloud computing

Comme mentionné précédemment « chapitre 2 », WSDL fournit des informations sur la façon d'interagir avec un service de manière fonctionnelle et technique, telles que les opérations, les entrées, les sorties et les paramètres par défaut. Un document WSDL utilise les éléments suivants pour décrire un service web (Types, Message, Operation, PortType, Binding, Port et Service).
Dans notre proposition, nous avons conservé ces éléments de base comme le montre la figure III.14 et la figure III.15.

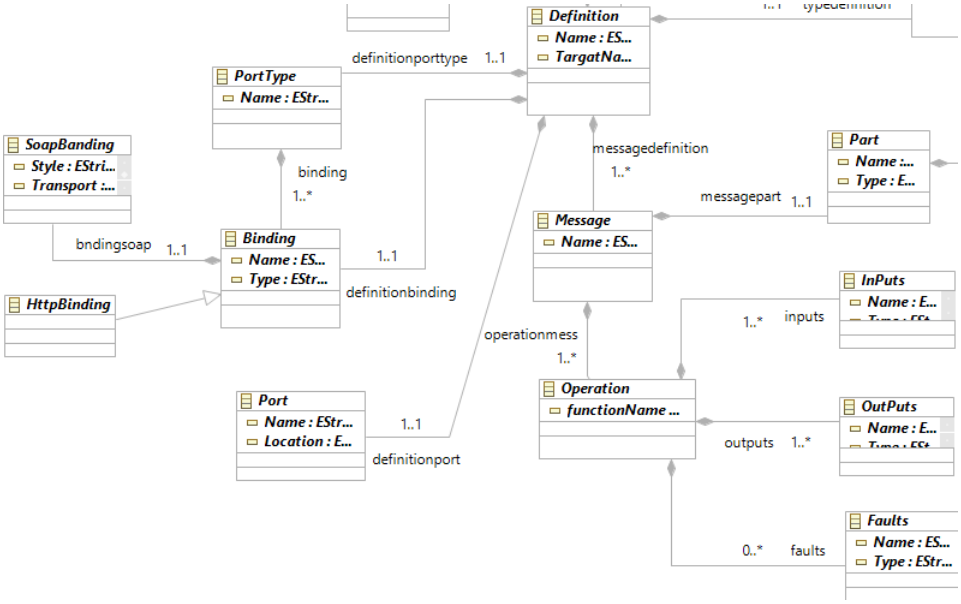

FIGURE IV.14 – Les notions de description service WSDL

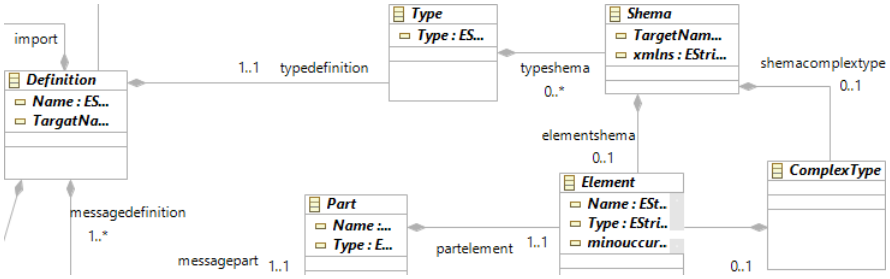

FIGURE IV.15 – Les notions de description service WSDL 2

Le WSDL ne peut pas prendre en charge la sémantique ni décrire les propriétés non fonctionnelles. De plus, WSDL ne peut pas permettre la notation des prix, l'évaluation, l'aspect juridique, le SLA, etc. Donc, dans notre proposition, nous avons ajouté les notions nécessaires pour adapter le WSDL au Cloud, les notions ajoutées sont :

- **Prix (pricing) :** wsdl doit inclure les différentes fonctionnalités liées à divers aspects de tarification. « Voir la figure III.16 ».
- **Juridique (Legal) :**le wsdl doit faire l'objet d'une licence. La licence inclut le droit d'utilisation. Toutes ces propriétés aident les utilisateurs à sélectionner facilement les services appropriés. « Voir la figure III.16 ».
- **Acteurs (Participant) :**Inclut les acteurs nécessaires pour le Cloud Computing côté organisation, client et agent. « Voir la figure III.16 ».

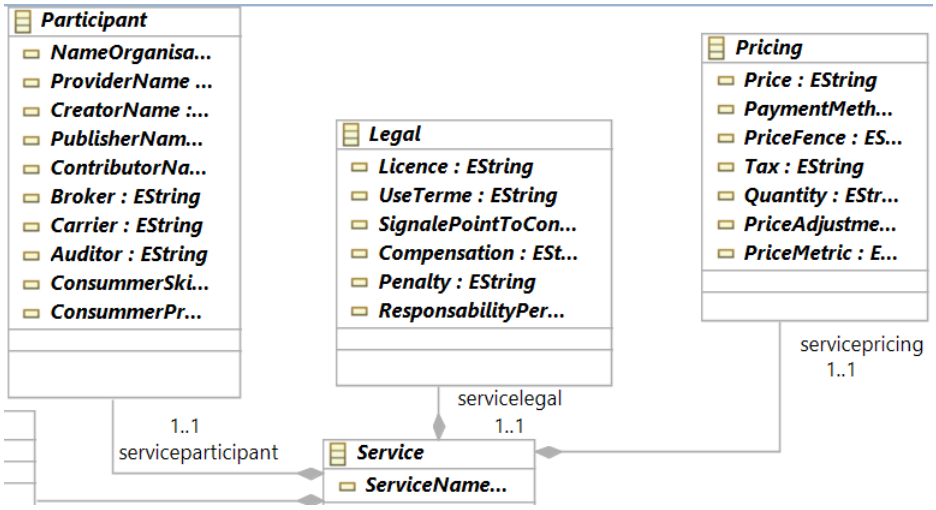

FIGURE IV.16 – Les notions ajoutées à la description service WSDL

- **Service level agreement (SLA) :** : nous avons ajouté pour le wsdl la notion de SLA pour les 3 types (SaaS, PaaS et IaaS).

  Pour le SaaS, le SLA contient les informations suivantes :rapports, temps de réponse, temps attendant de réparer et Disponibilité.

  Pour le PaaS, le SLA contient les informations suivantes : Gestion de protocole, Erreur code, Code de réussite, Incident, Ressource de service et Disponibilité.

  Pour le IaaS, le SLA contient les informations suivantes : Taux d'erreur, Moment de l'opération, Temps de service et Temps d'indisponibilité. « Voir la figure III.17 ».

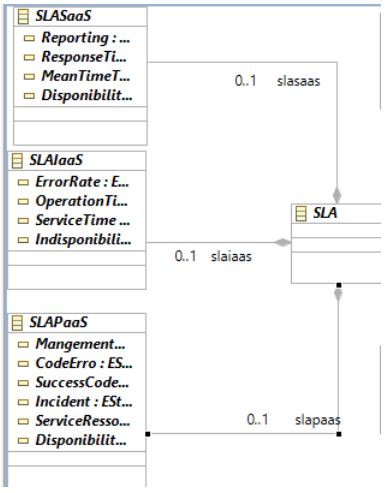

FIGURE IV.17 – Les notions de SLA ajoutées à la description service WSDL

- **Notion de cloud :**Dans cette mesure, nous avons ajouté au wsdl tout ce qui concerne le cloud :
  - Modèle de déploiement (Public, Privé, Hybride, Communauté).
  - Modèle de livraison (SaaS, PaaS, IaaS).
  - Évaluation des services (confiance, risque, réputation).
  - Expérience .
  - Estimation du consommateur.
  - Temps moyen de réparation, Temps de réponse et la disponibilité. « Voir la figure III.18 ».
- **Interaction :**qui se compose de simple interaction protocole et complexe interaction protocole. « Voir la figure III.18 ».

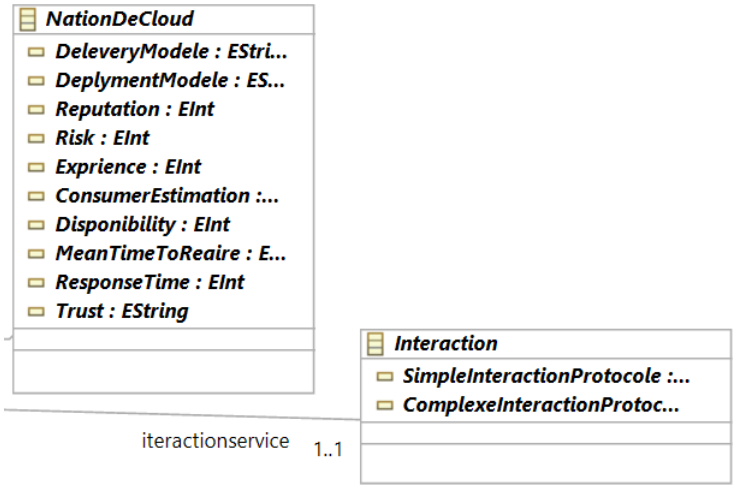

FIGURE IV.18 – Les notions de SLA ajoutées à la description service WSDL

- **Fondation :**regroupe les notions suivantes (Certificat, Expression et La description). « Voir la figure III.19 ».

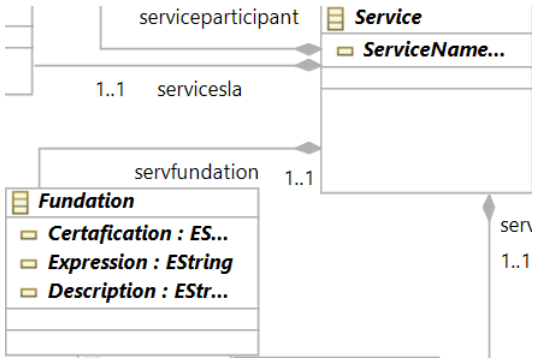

FIGURE IV.19 – Les notions de Fondation ajoutées à la description service WSDL

IV.4.2 Adaptation de USDL au cloud computing

Le langage de description de service unifié (USDL) est un langage permettant de décrire les services Internet sous trois angles différents (technique, commercial et opérationnel) « voir le chapitre 2 ».

Au début, USDL a été utilisé pour décrire des services Internet généraux, mais il n'est pas adapté à représenter les services cloud car il existe différentes caractéristiques qui caractérisent le cloud computing. Comme USDL est un langage indépendant de la plate-forme, il doit être redéfini pour prendre en compte ces fonctionnalités.

USDL est divisé en plusieurs paquets (selon la terminologie UML). Chaque paquet représente un « module » les différents modules USDL : Module de service, Module fonctionnel, Module technique, Module Participant, Module d'interaction, Module de prix, Module de niveau de service, Module juridique et Module Fondation.

Dans notre proposition, nous avons ajouté pour chaque module les notions nécessaires pour le cloud.

- **Module de service :**
  - Nom de service.
  - Modèle de déploiement (Public, Privé, Hybride, Communauté).
  - Modèle de livraison (SaaS, PaaS, IaaS).
  - Évaluation des services (confiance, risque, réputation).
  - Expérience.

- Estimation du consommateur.
- Temps moyen de réparation, Temps de réponse et la disponibilité.« voir la figure III.20 »

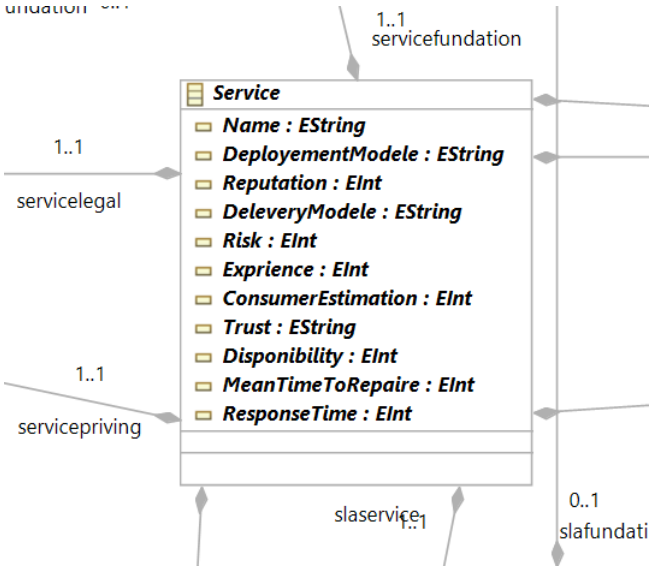

FIGURE IV.20 – Les notions de service ajoutées à la description service USDL

- **Module fonctionnel :** inclut les fonctions des services et leurs paramètres d'entrée/sortie et paramètre en cas des erreurs. « Voir la figure III.21».
- **Module technique :** contient le protocole de transport, URI de service, la localisation physique de service car le cloud nécessite la localisation physique et finalement les ressources qui regroupent les URI nécessaires pour la description. « Voir la figure III.21».
- **Module d'interaction :** se compose de simple interaction protocole et complexe interaction protocole. « Voir la figure III.21».

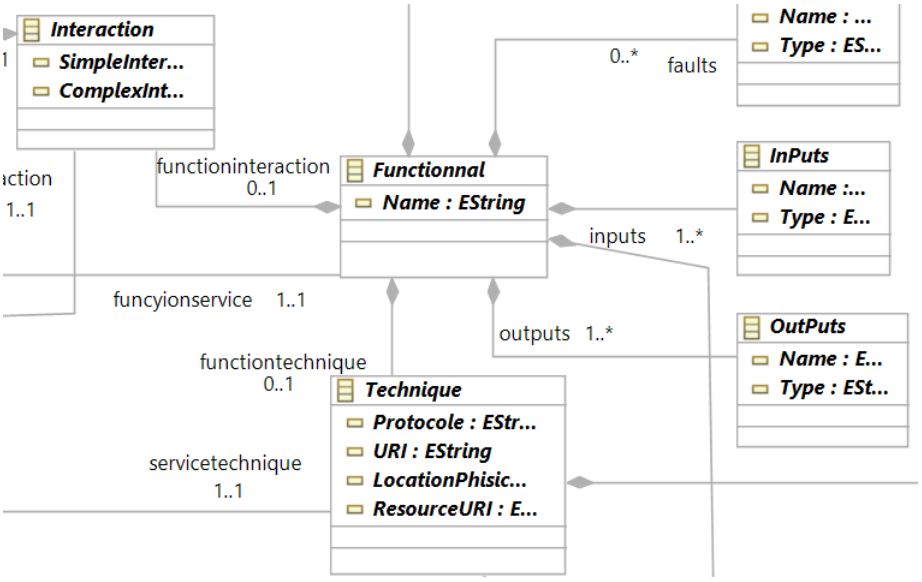

FIGURE IV.21 – Les notions de fonctionnel, technique et d'interaction ajoutées à la description service USDL

- **Module Participant :** rassemblement de toutes les personnes qui jouent un rôle dans le cloud, qu'elle soit affiliée à l'organisation, client ou associé au service. « Voir la figure III.22 ».
- **Module de prix :** Contient des concepts liés au prix, au paiement et aux transactions commerciales« Voir la figure III.22 ».

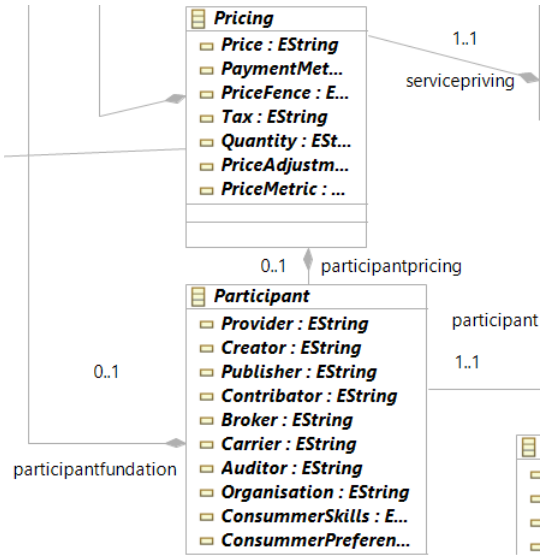

FIGURE IV.22 – Les notions de participant et prix ajoutées à la description service USDL

- **Module de niveau de service :** le USDL contient le module de SLA mais pas en détail, donc dans notre proposition nous avons ajouté le SLA pour chaque type de service. « Voir la figure III.23 ».

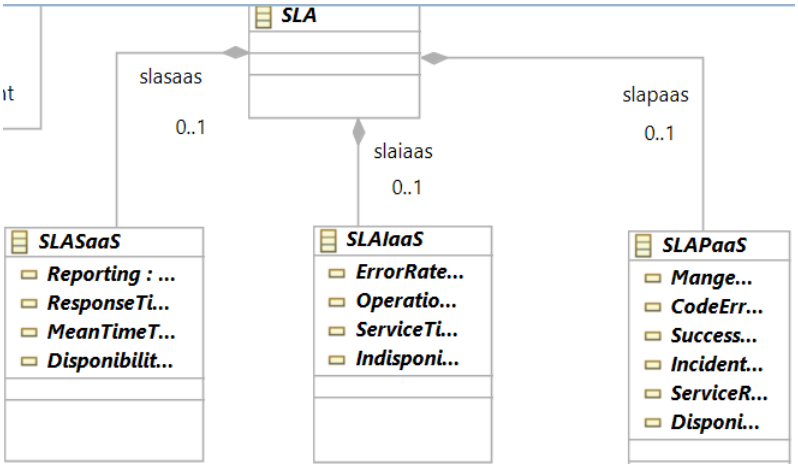

FIGURE IV.23 – Les notions de SLA ajoutées à la description service USDL

- **Module juridique :** Il contient les informations sur les conditions, les licences et les droits des utilisateurs pour les parties participantes. « Voir la figure III.24 ».
- **Module fondation :** Ce module fournit un ensemble de propriétés communes. Ces propriétés sont utilisées par tous les modules. « Voir la figure III.24 ».

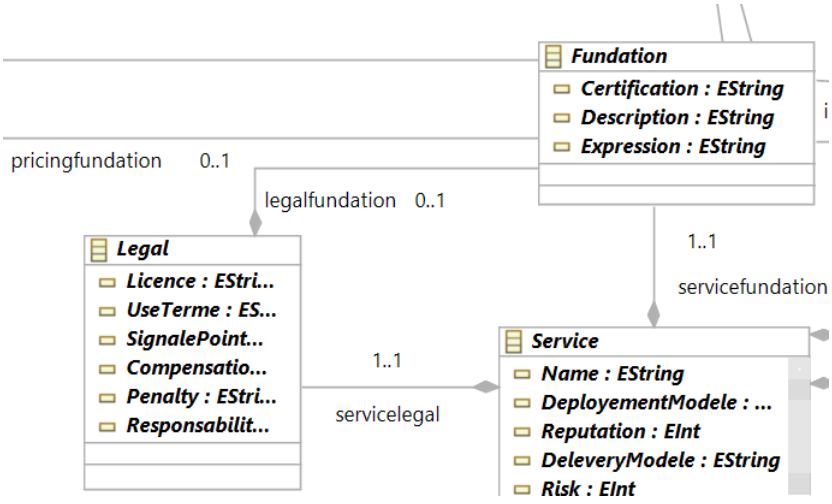

FIGURE IV.24 – Les notions de juridique et fondation ajoutées à la description service USDL

IV.4.3 Adaptation de OWL-S au cloud computing

OWL-S (Ontology Web Language for Services) est une ontologie de description des services web, dont les objectifs sont de résoudre les ambiguïtés et de rendre la description d'un service compréhensible par une machine « chapitre 2 ».
Le langage OWL-S, comme tous les autres langages de description, décrit le service en général, et il n'est pas destiné au cloud, pour cela, nous avons ajouté les éléments nécessaires à OWL-S pour le rendre adapté au cloud.
Le langage de description, comme nous l'avons mentionné précédemment, se compose de trois éléments de base que nous avons conservés avec l'ajout des éléments de base du Cloud pour devenir comme suit :

- **Service Profile :** qui indique "ce que fait le service"
  Dans cette partie, nous avons conservé les éléments de base du langage sans modifications ni ajouts. Le Service Profile se compose de :
  - **serviceName :** le nom du service offert.
  - **textDescription :** il résume ce que propose le service, il décrit ce dont le service a besoin pour fonctionner, et des informations supplémentaires.
  - **serviceClassification :** un mappage d'un profil à une ontologie OWL de prestations de service.
  - **contactInformation :** un mécanisme de référence aux humains ou aux individus responsables du service.
  - **CategoryName :** décrit des catégories de services sur la base d'une certaine classification (ontologie ou taxonomie).
  - **Taxonomy :** stocke une référence au schéma de taxonomie. Elle peut être soit une URI de la taxonomie, soit une URL où réside la taxonomie, ou le nom de la taxonomie ou autre.
  - **Value :** indique la valeur dans une taxonomie spécifique, il peut y avoir plus d'une valeur pour chaque taxonomie, donc aucune restriction n'est ajoutée ici.
  - **Code :** pour chaque type de service, stocke le code associé à une taxonomie.
  - **ServiceParameter :** il pointe vers la valeur du paramètre dans l'ontologie OWL. « Voir la figure III.25 ».

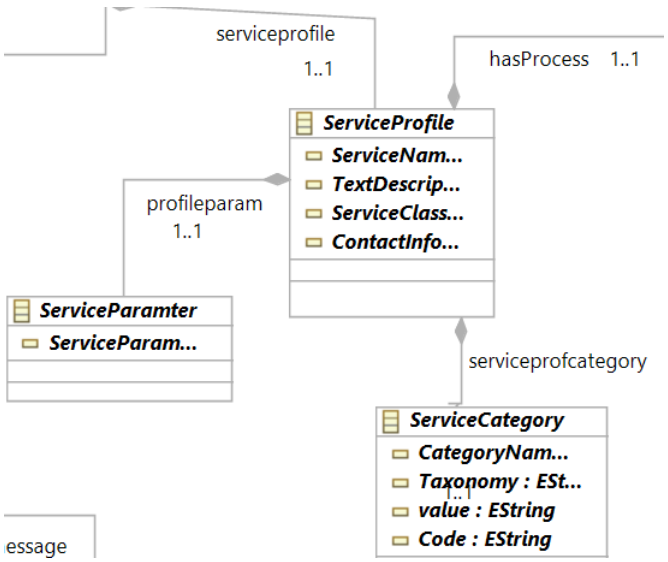

FIGURE IV.25 – Les notions de Service Profile de description service OWL-S

- **Service Process :** qui indique «comment le service fonctionne"

  Dans cette partie, nous avons également conservé les éléments de base du Service Process, mais nous avons fait quelques ajouts nécessaires pour rendre le langage de description compatible avec le cloud.

  Ces ajouts sont représentés dans ajout de legal dans la partie condition, ainsi que de la partie des participants nous avons ajouté l'élément organisation et mentionné les participants importants dans le cloud (client, service). Le Service Process devient comme suit :

  - **Result :** il spécifie quels changements de domaine sont produits pendant l'exécution de la prestation. « Voir la figure III.26 ».
  - **hasResult :** spécifie un des résultats du service. « Voir la figure III.26».
  - **Condition :** contient expression et légal liés au cloud. « Voir la figure III.26».

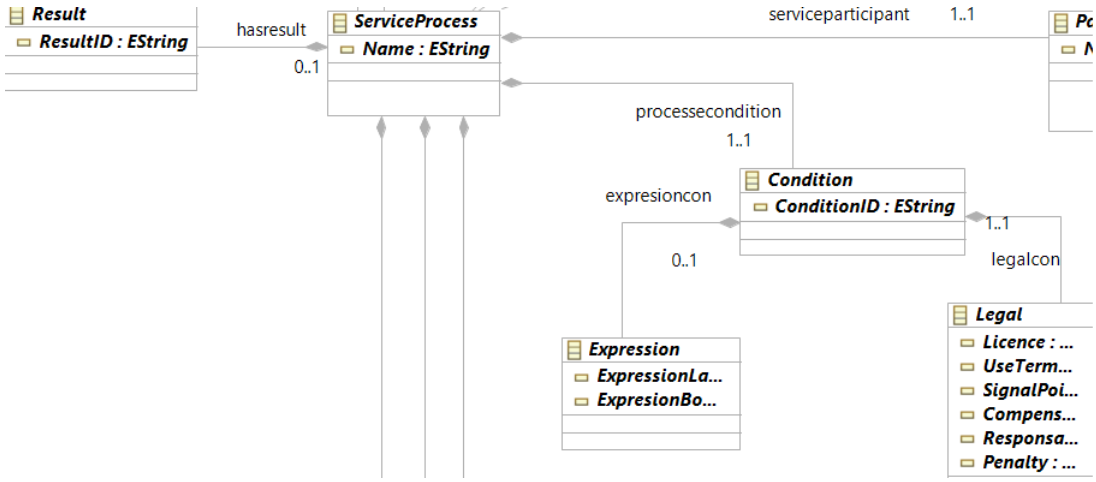

FIGURE IV.26 – Les notions deresult, hasResult et condition de description service OWL-S

- **Participant :** les agents qui sont impliqués dans le processus. « Voir la figure III.27».
- **The client :** l'agent du point de vue duquel le processus est décrit. « Voir la figure III.27».
- **The service :** l'élément principal du service que le client rend. « Voir la figure III.27».

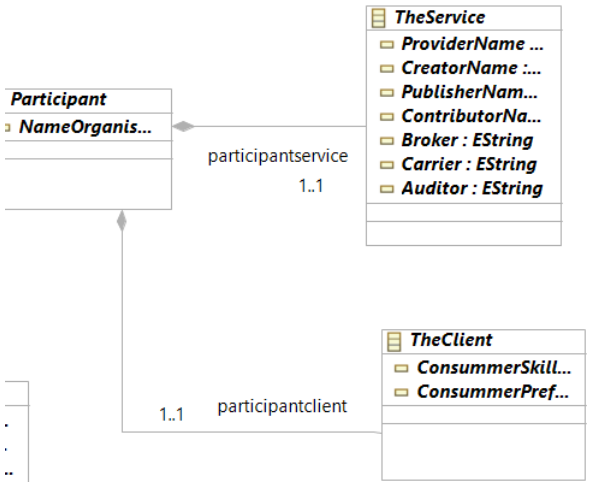

FIGURE IV.27 – Les notions de participant de description service OWL-S

- **Processus composite :** maintient un certain état, chaque message envoyé par le client le fait avancer à travers le processus. « Voir la figure III.28».
- **Processus simple :** utilisé pour l'abstraction et la simplification. « Voir la figure III.28».
- **Processus atomique :** une description d'un service qui attend un (éventuellement complexe) message et en renvoie un message (éventuellement complexe). « Voir la figure III.28».
- **Séquence :** une liste de constructions de contrôle à faire dans l'ordre. « Voir la figure III.28».
- **Split :** les composants d'un processus Split sont un sac de composants de processus à exécuter simultanément. La division se termine dès que les processus composants ont été planifiés pour exécution. « Voir la figure 27».
- **Choise :** demande l'exécution d'un contrôle unique à construire à partir d'un sac donné de constructions de contrôle (donné par la propriété des composants). N'importe laquelle des constructions de contrôle données peut être choisie pour l'exécution. « Voir la figure III.28».
- **SplitLion :** ici, le processus consiste en des opérations simultanées exécutées d'un tas de composants des processus avec synchronisation. « Voir la figure III.28».

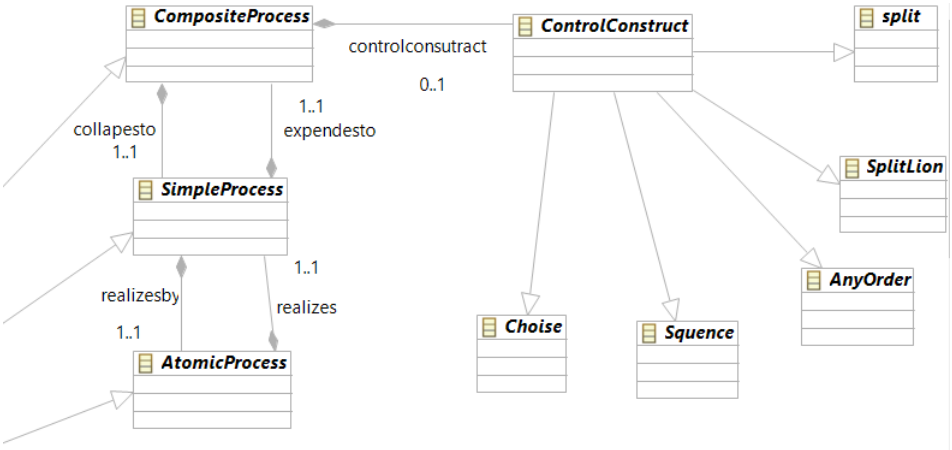

FIGURE IV.28 – Les notions de Processus composite de description service OWL-S

- **Hasinputs, hasoutputs, hasfaults :** lier le service process avec les paramètres entrée sortie et fault dans la partie grounding.

- **Service Grounding :** qui indique ”comment accéder au service”

  Comme les autres parties, nous avons gardé les éléments de base et ajouté ce qui était nécessaire pour le cloud où nous avons ajouté PhysicalLocation et Interaction.

  Dans Service Grounding on trouve :

  - Un protocole de communication.
  - Formats de messages.
  - Autres détails spécifiques au service tels que les numéros de port utilisés en contactant le service et les fonctions du service et leurs paramètres. « Voir la figure III.29».

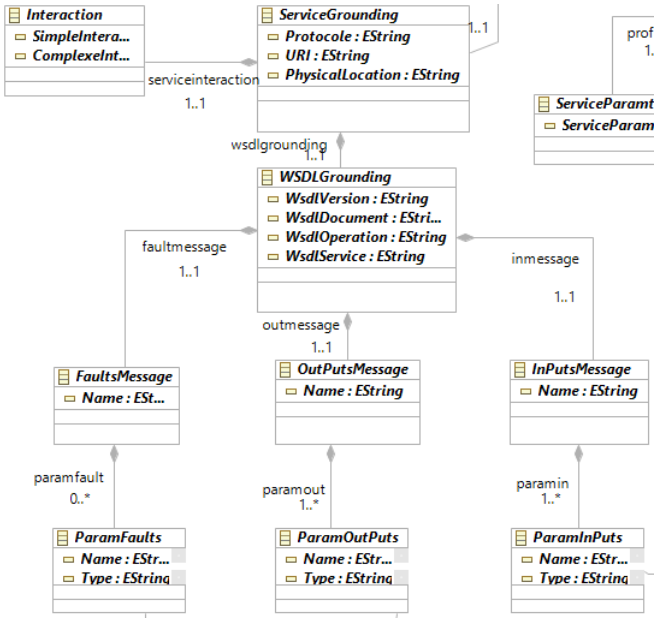

FIGURE IV.29 – Les notions de service grounding de description service OWL-S

- Ce sont les éléments qui composaient le langage de description et pour le rendre compatible avec le cloud nous avons également ajouté SLA, NotionDeCloud, pricing et Fundation. « Voir la figure III.30».

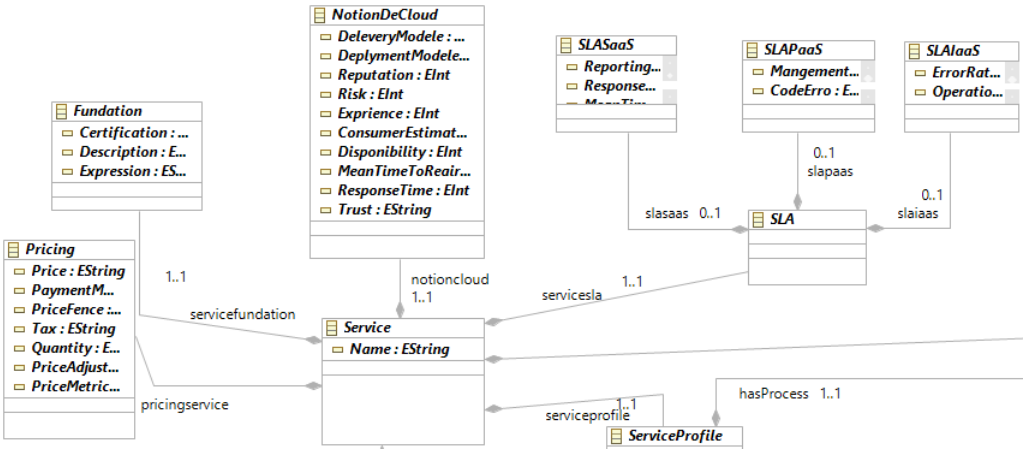

FIGURE IV.30 – Les notions de cloud de description service OWL-S

IV.4.4 Adaptation de WSMO au cloud computing

WSMO est un langage de description pour les services Web sémantiques « chapitre 2 ». Ce langage de description est très faibles en termes d'informations dont le service a besoin. Nous avons donc conservé ses éléments et y avons ajouté ce qui est nécessaire pour le cloud.
Les éléments de base du WSMO-lite sont « Voir la figure III.31» :

- **OntologyElement :** les ressources des ontologies.
- **Transformation :** le protocole de transformation, URI et nous avons ajouté la localisation physique. Nous trouvons aussi le type de transformation « lifting, lowering ».
- **Operation :** les fonctions de service et leurs paramètres.
- **FunctionalCategory.**
- **NonFunctionalparamter .**

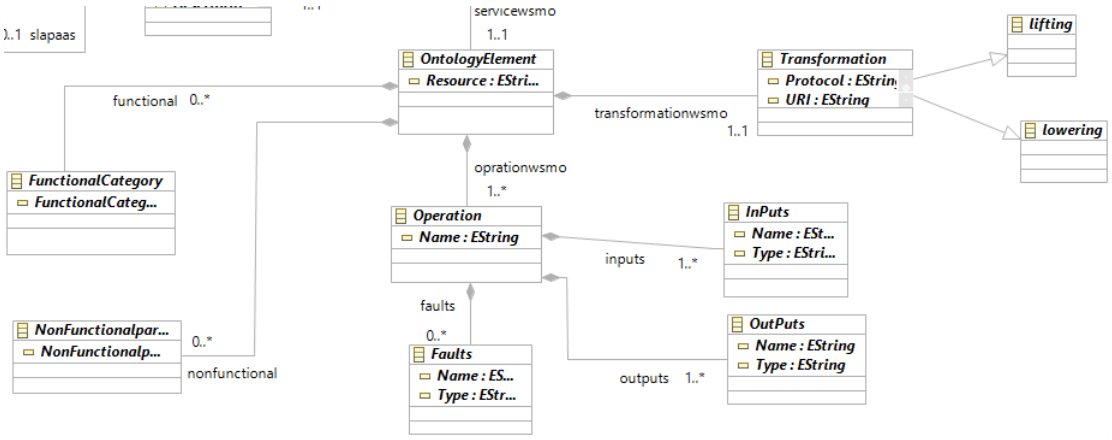

FIGURE IV.31 – Les éléments de base de description service WSMO

En plus des éléments de base, nous avons ajouté les éléments du cloud représentés par : Participant, Legal, Pricing, Interaction, NotionDeCloud, Fundation et SLA. « Voir la figure III.32».

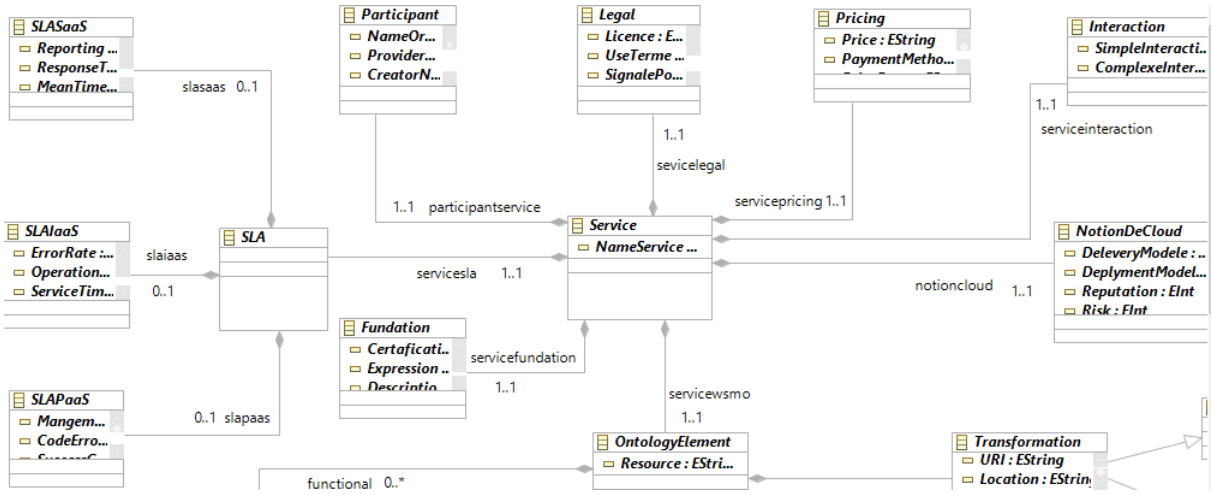

FIGURE IV.32 – Les notions de cloud de description service WSMO

IV.5 Les règles de transformation :

Après avoir fourni un langage de description général qui inclut tous les concepts nécessaires du cloud, et après les modifications que nous avons apportées aux langages de description les plus courants (WSDL, OWL-s, USDL et WSMO-lite) pour les rendre compatibles avec le langage de description, nous passons à l'étape suivante, qui consiste à convertir différentes descriptions dans le langage de description du cloud.

Nous nous appuyons sur Atlas Transformation Language (ATL) pour mettre en œuvre les règles de transformation. ATL est un langage spécifique à un domaine pour les transformations unidirectionnelles de modèle à modèle (M2M), qui fournit des constructions déclaratives et impératives.

Dans cette partie, nous présentons notre transformation en utilisant ATL pour transformer WSDL, OWL-s, USDL et WSMO-lite vers le langage de description de cloud, et également convertir dans le sens opposé afin qu'il n'y ait aucune limitation sur le langage de description utilisé pour éviter le problème de verrouillage du fournisseur.

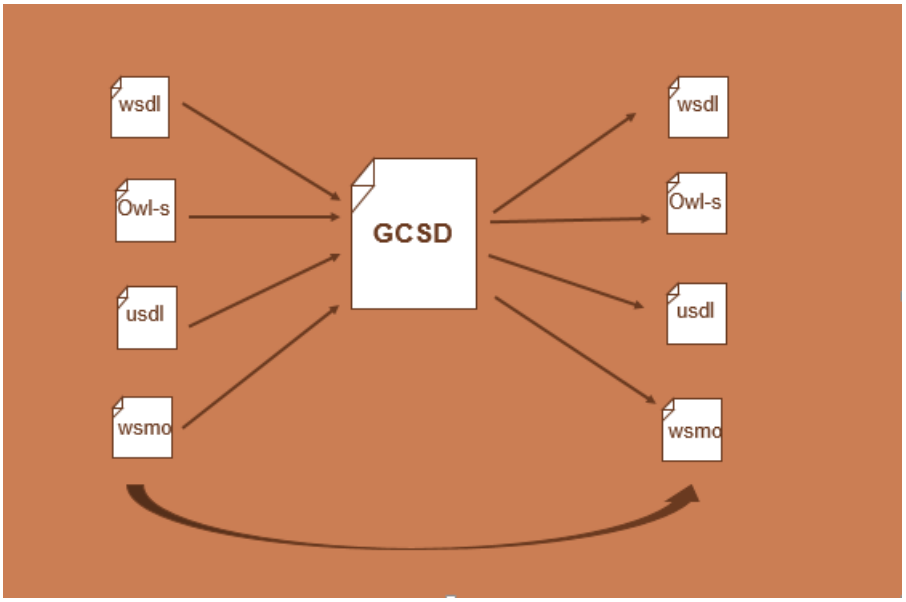

FIGURE IV.33 – Transformation des langages

IV.5.1 Règles de transformation de WSDL vers le GCSD

La figure III.34, illustre la mise en œuvre des règles de transformation de WSDL à GCSD.

```
module WSDL2GCSD;
create OUT : gcsd from IN : wsdl;

rule service2ServiceEvulition {
    from
        s : wsdl!NationDeCloud,
        sl :wsdl!Service
    to
        t : gcsd!Service (NameOfService <- sl.ServiceName),
        t1 : gcsd!DeployementModel (TypeDeplyement <- s.DeplymentModele),
        t2 : gcsd!DeleveryModele ( DeleveryModele <- s.DeleveryModele),
        t3 :gcsd!Risk ( Risk <- s.Risk),
        t4 : gcsd!Reputation ( Reputation <- s.Reputation),
        t5 : gcsd!Trust ( Trust <- s.Trust)
}
rule wsdl2contex {
    from
        s : wsdl!NationDeCloud,
        sl : wsdl!Definition
    to
        t : gcsd!Exprience (Exprience <- s.Exprience),
        t1 : gcsd!ConsumerEstimation ( ConsumerEstimation <- s.ConsumerEstimation),
        t2 : gcsd!ReponceTime ( ReponceTime <- s.ResponseTime),
        t3 : gcsd!MeanTimeToRepaire( MeanTimeToRepaire <- s.MeanTimeToReaire),
        t4 : gcsd!Disponibility ( Disponibility <- s.Disponibility),
        t5 : gcsd!Resource( URIRussource <- sl.TargatNameSpace)
}
```

FIGURE IV.34 – Transformation de WSDL vers GCSD

Pour la transformation de WSDL vers le GCSD, nous avons utilisé les règles de transformation suivantes :

- ***service2ServiceEvulition :***transforme le service WSDL qui contient le nom de service au nom de service dans le GCSD de partie service, aussi la transformation d'évolution qui est inclue dans la notion de cloud de WSDL au service évolution dans le GCSD, et aussi la transformation de type déploiement et livraison.
- ***wsdl2contex :***transforme les restes de notions de cloud qui existent dans wsdl au contex de GCSD, aussi la transformation de trgat name space dans la partie définition de WSDL à la ressource de partie contex de GCSD.
- ***wsdl2sla :***responsable de la conversion de SLA pour chaque type (PaaS, IaaS et SaaS) trouvé dans WSDL à sa pertinence dans GCSD.
- ***wsdl2pricing :***transforme les éléments de pricing dans WSDL aux éléments de pricing dans GCSD.
- ***wsdl2participant :***transforme les participants de WSDL au GCSD, commençant par le client, on trouve la transformation de ConsumerPreferences et ConsumerSkills, après la transformation de service qui contient la transformation de Provider, Creator, Publisher, Contributor, Broker, Carrier et Auditor. Au final, la transformation de nom d'organisation.
- ***wsdl2fundation :*** responsable de la conversion de fundation trouvé dans WSDL vers Certification, Expression Description dans le GCSD.
- ***wsdl2legal :***transforme les notions de legal de WSDL au legal de GCSD.
- ***wsdl2technique :***responsable du transport de service. Dans cette règle, on trouve la conversion de protocole de transport qui trouve dans SoapBanding de WSDL au protocole qui se trouve dans la partie technique dans GCSD, aussi on trouve la transformation de location de service qui se trouve dans la partie Port de WSDL au URI de GCSD qui se trouve dans technique.
- ***wsdl2function :***transforme l'opération WSDL en concept Functionnal. L'opération, les propriétés d'entrée, de sortie et de défaut sont décrites respectivement par les relations GCSD entrées, sorties et défauts, et le nom de l'opération est décrit par la propriété functionName.
- ***wsdl2iteraction :***transforme iteraction WSDL au iteration GCSD.

IV.5.2 Règles de transformation d'OWL-S vers le GCSD

La figure III.35 illustre la mise en œuvre des règles de transformation de OWL-S à GCSD.

```
module OWLS2GCSD;
create OUT : gcsd from IN : owls;

rule owls2service {
    from
        s : owls!NotionDeCloud,
        sl : owls!Service
    to
        t : gcsd!Service (NameOfService <- sl.Name),
        t1 : gcsd!DeployementModel( TypeDeplyement <- s.DeplymentModele),
        t2 :gcsd!DeleveryModele ( DeleveryModele <- s.DeleveryModele),
        t3 :gcsd!Risk ( Risk <- s.Risk),
        t4: gcsd!Reputation ( Reputation <- s.Reputation),
        t5 : gcsd!Trust (Trust <- s.Trust)
}
rule owls2contex {
    from
        s : owls!NotionDeCloud,
        sl : owls!ServiceProfile
    to
        t : gcsd!Exprience (Exprience <- s.Exprience),
        t1 :gcsd!ConsumerEstimation( ConsumerEstimation <- s.ConsumerEstimation),
        t2 :gcsd!ReponceTime ( ReponceTime <- s.ResponseTime),
        t3 :gcsd!MeanTimeToRepaire( MeanTimeToRepaire <- s.MeanTimeToReaire),
        t4 : gcsd!Disponibility ( Disponibility <- s.Disponibility),
        t5 :gcsd!Resource (URIRussource <- sl.ServiceClassification)

}
```

FIGURE IV.35 – Transformation de OWL-S vers GCSD

Pour la transformation de OWL-S vers le GCSD, nous avons utilisé les règles de transformation suivantes :

- ***owls2service :***transforme le service OWL-S qui contient le nom de service au nom de service dans le GCSD de partie service, aussi la transformation d'évolution qui est inclue dont la notion de cloud d'OWL-S au service évolution dans le GCSD, et aussi la transformation de type déploiement et livraison.
- ***owls2contex :***transforme les restes de notions de cloud qui existent dans OWL-S au contex de GCSD, aussi la transformation de ServiceClassification qui contient le URI de ontologie dans la partie ServiceProfile de OWL-S à la ressource de partie contex de GCSD.
- ***owls2sla :***responsable de la conversion de SLA pour chaque type (PaaS, IaaS et SaaS) trouvé dans OWL-S à sa pertinence dans GCSD.
- ***owls2pricing :***transforme les éléments de pricing dans OWL-S aux éléments de pricing dans GCSD.
- ***owls2participant :***transforme les participants de OWL-S au GCSD , commençant par le client, on trouve la transformation de élément TheClient de OWL-S au ConsumerPreferences et ConsumerSkills de GCSD, après la transformation de service qui se trouve dans TheService de OWL-S au Provider, Creator, Publisher, Contributor, Broker, Carrier et Auditor de GCSD. Au final, la transformation de nom d'organisation qui se trouve dans Participant de OWL-S.
- ***owls2fundation :***responsable de la conversion de fundation trouvé dans OWL-S à Certification, Expression Description dans le GCSD.
- ***owls2legal :*** transforme les notions de legal de OWL-S au legal de GCSD.
- ***owls2technique :*** on trouve la conversion de protocole de transport qui se trouve dans ServiceGrounding de OWL-S au protocole qui se trouve dans la partie technique dans GCSD, aussi on trouve la transformation de location de service et location physique au URI de GCSD qui se trouve dans technique et PhysicalLocation qui se trouve dans location.
- ***owls2functional :***transforme WSDLGrounding qui contient le nom de l'opération « WsdlOperation » en nom de fonction dans GCSD. Les propriétés d'entrée, de sortie et de défaut et leur type sont décrites respectivement par les relations GCSD entrées, sorties et défauts.
- ***owls2iteraction :***transforme iteraction OWL-S à iteration GCSD.

IV.5.3 Règles de transformation d'USDL vers le GCSD

La figure III.36 illustre la mise en œuvre des règles de transformation d'USDL à GCSD.

```
module USDL2GCSD;
create OUT : gcsd from IN : usdl;


rule service2ServiceEvulition {
    from
        s : usdl!Service
    to
        t3 : gcsd!Service ( NameOfService <- s.Name),
        t4 : gcsd!DeployementModel (TypeDeplyement <- s.DeleveryModele),
        t5 : gcsd!DeleveryModele (DeleveryModele <- s.DeleveryModele),
        t : gcsd!Risk (Risk <- s.Risk),
        t1 : gcsd!Reputation (Reputation <- s.Reputation),
        t2 : gcsd!Trust (Trust <- s.Trust)

}
rule usdl2contex {
    from
        s : usdl!Service,
        s1 : usdl!Technique
    to
        t : gcsd!Exprience (Exprience <- s.Exprience),
        t1 : gcsd!ConsumerEstimation (ConsumerEstimation <- s.ConsumerEstimation),
        t2 : gcsd!ReponceTime (ReponceTime <- s.Reputation),
        t3 : gcsd!MeanTimeToRepaire (MeanTimeToRepaire <- s.MeanTimeToRepaire),
        t4 : gcsd!Disponibility (Disponibility <- s.Disponibility),
        t5 : gcsd!Resource (URIRussource <- s1.ResourceURI)

}
```

FIGURE IV.36 – Transformation de USDL vers GCSD

Pour la transformation de USDL vers le GCSD, nous avons utilisé les règles de transformation suivantes :

- ***service2ServiceEvulition :***transforme le service USDL qui contient le nom de service au nom de service dans le GCSD de partie service, aussi la transformation d'évolution qui est inclue dans service d'USDL au service évolution dans le GCSD, et aussi la transformation de type déploiement et livraison.
- ***usdl2contex :***transforme les restes de notions de cloud qui existent dans service USDL au contex de GCSD, aussi la transformation de ResourceURI qui contient les URI des ressources USDL dans la partie technique à la ressource de partie contex de GCSD.
- ***usdl2sla :***responsable de la conversion de SLA pour chaque type (PaaS, IaaS et SaaS) trouvé dans USDL à sa pertinence dans GCSD.
- ***usdl2pricing :***transforme les éléments de pricing dans USDL aux éléments de pricing dans GCSD.
- ***usdl2participantclient :***transforme les participants d'USDL au GCSD, commençant par le client, on trouve la transformation de élément ConsumerPreferences et ConsumerSkills, après la transformation de service au Provider, Creator, Publisher, Contributor, Broker, Carrier et Auditor de GCSD. Au final, la transformation de nom d'organisation.
- ***usdl2fundation :*** responsable de la conversion de fundation trouvé à USDL au Certification, Expression Description dans le GCSD.
- ***usdl2legal :***transforme les notions de légal de USDL au legal de GCSD.
- ***usdl2technique :***on trouve la conversion de protocole de transport qui se trouve dans technique d'USDL au protocole qui se trouve dans la partie technique dans GCSD, aussi on trouve la transformation de location de service au URI de GCSD qui se trouve dans technique.
- ***usdl2function :***transforme nom de la fonction USDL en nom de fonction dans GCSD. Les propriétés d'entrée, de sortie et de défaut et leurs types sont décrites respectivement par les relations GCSD entrées, sorties et défauts.
- ***usdl2iteraction :***transforme iteraction USDL au iteration GCSD.

IV.5.4 Règles de transformation de WSMO vers le GCSD

La figure III.37 illustre la mise en œuvre des règles de transformation de WSMO à GCSD.

```
module WSMO2GCSD;
create OUT : gcsd from IN : wsmo;

rule wsmo2service {
    from
        s : wsmo!Service,
        s1: wsmo!NotionDeCloud
    to
        t : gcsd!Service (NameOfService <- s.NameService),
        t1:gcsd!DeployementModel ( TypeDeplyement <- s1.DeplymentModele),
        t2 : gcsd!DeleveryModele ( DeleveryModele <- s1.DeleveryModele),
        t3 : gcsd!Risk ( Risk <- s1.Risk),
        t4 : gcsd!Trust (Trust <- s1.Trust),
        t5 : gcsd!Reputation ( Reputation <- s1.Reputation)

}
rule wsmo2contex {
    from
        s : wsmo!NotionDeCloud,
        s1: wsmo!OntologyElement
    to
        t : gcsd!Exprience (Exprience <- s.Exprience),
        t1 :gcsd!ConsumerEstimation( ConsumerEstimation <- s.ConsumerEstimation),
        t2 :gcsd!ReponceTime ( ReponceTime <- s.ResponseTime),
        t3 : gcsd!MeanTimeToRepaire ( MeanTimeToRepaire <- s.MeanTimeToReaire),
        t4 : gcsd!Disponibility ( Disponibility <- s.Disponibility),
        t5 :gcsd!Resource ( URIRussource <- s1.Resource)
}
```

FIGURE IV.37 – Transformation de WSMO vers GCSD

Pour la transformation de WSMO vers le GCSD, nous avons utilisé les règles de transformation suivantes :

- ***wsmo2service :***transforme le nom service WSMO au nom de service dans le GCSD, aussi la transformation d'évolution qui est inclue dans notion de cloud de WSMO au service évolution dans le GCSD, et aussi la transformation de type déploiement et livraison.
- ***wsmo2contex :***transforme les restes de notions de cloud qui existent dans WSMO au contex de GCSD, aussi la transformation de Resource qui contient les URI de ontologie dans la partie OntologyElement à la ressource de partie contex de GCSD.
- ***wsmo2sla :***responsable de la conversion de SLASaaS, SLAIaaS et SLAPaaS trouvé dans WSMO à sa pertinence dans GCSD.
- ***wsmo2pricing :***transforme les éléments de pricing dans WSMO aux éléments de pricing dans GCSD.
- ***wsmo2participant :***transforme les participants de WSMO au GCSD, commençant par le client, on trouve la transformation de élément ConsumerPreferences et ConsumerSkills,

après la transformation de service au Provider, Creator, Publisher, Contributor, Broker, Carrier et Auditor de GCSD. Au final, la transformation de nom d'organisation.

- ***wsmo2fundation :***responsable de la conversion de fundation trouvé dans WSMO à Certification, Expression Description dans le GCSD.
- ***wsmo2legal :***transforme les notions de légal de WSMO au legal de GCSD.
- ***wsmo2technique :***on trouve la conversion de protocole de transport et URI de service qui se trouve dans Transformation de WSMO au protocole et URI qui se trouve dans la partie technique dans GCSD.
- ***wsmo2function :***transforme nom de l'operation WSMO en nom de fonction dans GCSD. Les propriétés d'entrée, de sortie et de défaut et leur type sont décrites respectivement par les relations GCSD entrées, sorties et défauts.
- ***wsmo2iteraction :***transforme iteraction USDL à iteration GCSD.

IV.5.5 Règles de transformation du GCSD vers le WSDL

Dans cette étape, nous faisons la transformation inverse de ce que nous avons fait précédemment, la figure III.38 montre la transformation inverse du langage de description GCSD vers le WSDL.

```
module GCSD2WSDL;
create OUT : wsdl from IN : gcsd;

rule gcsd2servicewsdl {
    from
        s : gcsd!Service
    to
        t : wsdl!Service (ServiceName <- s.NameOfService)
}

rule gcsd2notioncloudwsdl {
    from
        s : gcsd!DeployementModel,
        s1 : gcsd!DeleveryModele,
        s2 : gcsd!Risk,
        s3 : gcsd!Trust,
        s4 : gcsd!Reputation,
        s5 :gcsd!Exprience,
        s6 : gcsd!ConsumerEstimation,
        s7: gcsd!Disponibility,
        s8 : gcsd!MeanTimeToRepaire,
        s9 : gcsd!ReponceTime
    to
        t : wsdl!NationDeCloud (DeplymentModele <- s.TypeDeplyement, DeleveryModele <- s1.DeleveryModele,Risk <- s2.Risk,
            Trust <- s3.Trust, Reputation <- s4.Reputation, Exprience <- s5.Exprience, ConsumerEstimation <- s6.ConsumerEstimation,
            Disponibility <- s7.Disponibility, MeanTimeToReaire <- s8.MeanTimeToRepaire, ResponseTime <- s9.ReponceTime

        )
}
```

FIGURE IV.38 – Transformation de GCSD vers WSDL

Dans cette transformation, nous utilisons les règles suivantes :

- ***gcsd2servicewsdl :*** qui transforme le nom de service.
- ***gcsd2notioncloudwsdl :*** qui transforme les notions de cloud.
- ***gcsd2fundation :*** qui transforme la fundation.
- ***gcsd2pricing :*** qui transforme le pricing.
- ***gcsd2legal :*** qui transforme legal.
- ***gcsd2interaction :*** qui transforme interaction.
- ***gcsd2participant :*** qui transforme les participants.
- ***gcsd2sla :*** qui transforme les SLA de 3 types.
- ***gcsd2definition :*** cette règle est responsable de la transformation de nom de service et URI de définition WSDL.
- ***gcsd2soapbinding :*** on trouve la transformation de protocole de GCSD au protocole de soapbanding de WSDL.
- ***gcsd2message :*** convertir le nom de message au WSDL à partir de functionnal de GCSD.
- ***ggcsd2operationMessage :*** ajouter le nom de fonction à l'opération et les paramètres, entrée sortie et faut à partir de inputs, outputs, faults et Functionnal de GCSD.
- ***gcsd2port :*** la transformation d'URI de service et le nom.

IV.5.6 Règles de transformation du GCSD vers le OWL-S

Dans cette étape, nous faisons la transformation inverse de ce que nous avons fait précédemment, la figure III.39 montre les transformations inverses du langage de description GCSD vers OWL-S.

```
module GCSD2OWLS;
create OUT : owls from IN : gcsd;

rule gcsd2owls {
    from
        s : gcsd!Service
    to
        t : owls!Service (Name <- s.NameOfService)
}
rule gcsd2notioncloudwsdl {
    from
        s : gcsd!DeployementModel,
        s1 : gcsd!DeleveryModele,
        s2 : gcsd!Risk,
        s3 : gcsd!Trust,
        s4 : gcsd!Reputation,
        s5 :gcsd!Exprience,
        s6 : gcsd!ConsumerEstimation,
        s7: gcsd!Disponibility,
        s8 : gcsd!MeanTimeToRepaire,
        s9 : gcsd!ReponceTime
    to
        t : owls!NotionDeCloud (DeleveryModele <- s.TypeDeplyement, DeleveryModele <- s1.DeleveryModele,Risk <- s2.
            Trust <- s3.Trust, Reputation <- s4.Reputation, Exprience <- s5.Exprience, ConsumerEstimation <- s6.Con
            Disponibility <- s7.Disponibility, MeanTimeToReaire <- s8.MeanTimeToRepaire, ResponseTime <- s9.Reponce
            )
}
```

FIGURE IV.39 – Transformation de GCSD vers OWL-S

Dans cette transformation, nous utilisons les règles suivantes :

- ***gcsd2owls :*** transformation de nom de service.
- ***gcsd2notioncloudwsdl :*** transformation des notions de cloud.
- ***gcsd2fundation :*** transformation de fundation (certification, expression et description).
- ***gcsd2pricing :*** transformation des notions de pricing.
- ***gcsd2condition :*** dans cette règle, on fait la transformation de legal quise trouve dans la partie condition d'OWL-S et la transformation d'expression à partir de fundation GCSD.
- ***gcsd2interaction :*** transformation d'interaction.
- ***gcsd2participant :*** transforme le nom d'organisation dans la partie participant d'OWL-S.
- ***gcsd2participantclient :*** cette règle transforme les agents clients de GCSD vers les agents clients d'OWL-S dans la partie theclient.
- ***gcsd2participantservice :*** cette règle transforme les agents de service de GCSD vers les agents de service d'OWL-S dans la partie theservice.
- ***gcsd2sla :*** transformation de SLA de type SaaS, IaaS et PaaS.
- ***gcsd2servicedefinition :*** cette règle transforme le nom de service à partir de service GCSD vers nom de service dans serviceprofile de OWL-S, et ajoute à serviceclassifica-

tion de OWL-S de partie serviceprofile URI de ontologie à partir de ressource GCSD, et on trouve aussi la transformation de description fundation GCSD vers description serviceprofile de OWL-S.

- ***gcsd2ServiceGrounding :*** transformation de protocole de transport et location physique et URI de service à travers technique GCSD vers servicegrounding d'OWL-S.
- ***gcsd2WSDLGrounding :*** transformation de nom de function à partir de functionnal de GCSD au WSDLOperation qui inclut le nom de fonction dans OWL-S.
- ***gcsd2paramin :*** la transformation de nom et type de paramètres d'entrées.
- ***gcsd2paramout :*** la transformation de nom et type de paramètres de sorties.
- ***gcsd2paramfaults :*** la transformation de nom et type de paramètres de fautes.

IV.5.7 Règles de transformation du GCSD vers le USDL

Dans cette étape, nous faisons la transformation inverse de ce que nous avons fait précédemment, la figure III.40 montre les transformations inverses du langage de description GCSD vers le USDL..

```
module GCSD2USDL;
create OUT : usdl from IN : gcsd;

rule gcsd2serviceusdl {
    from
        s : gcsd!Service,
        s1 : gcsd!DeployementModel,
        s2 : gcsd!DeleveryModele,
        s3 : gcsd!Risk,
        s4 :gcsd!Reputation,
        s5 :gcsd!Trust,
        s6 : gcsd!Exprience,
        s7 : gcsd!ConsumerEstimation,
        s8 :gcsd!Disponibility,
        s9 : gcsd!MeanTimeToRepaire,
        s10 : gcsd!ReponceTime
    to
        t : usdl!Service (Name <- s.NameOfService, DeleveryModele <- s2.DeleveryModele, DeployementModele <- s1.TypeDeplyement,
            Reputation <- s4.Reputation, Risk <- s3.Risk, Exprience <- s6.Exprience, ConsumerEstimation <- s7.ConsumerEstimation,
            Disponibility <- s8.Disponibility , MeanTimeToRepaire <- s9.MeanTimeToRepaire ,ResponseTime <- s10.ReponceTime

        )
}
rule gcsd2fundation {
    from
        s : gcsd!Certafication,
        s1 : gcsd!Description,
        s2 : gcsd!Expression
    to
        t : usdl!Fundation (Certification <- s.Certafication, Description <- s1.Description, Expression <- s2.Expression

        )
```

FIGURE IV.40 – Transformation de GCSD vers USDL

Dans cette transformation, nous utilisons les règles suivantes :

- ***gcsd2serviceusdl :*** la transformation de nom service et les notions de cloud.
- ***gcsd2fundation :*** : transformation de fundation (certification, description et expression).
- ***gcsd2pricing :*** transformation de pricing.
- ***gcsd2legal :*** transformation de legal.
- ***gcsd2interaction :*** transformation d'interaction.
- ***gcsd2participant :*** transformation des participants clients, services et organisation.
- ***gcsd2sla :*** transformation des paramètres SLA de 3 types.
- ***gcsd2technique :*** dans cette règle, on trouve la transformation de protocole de transport et location service et location physique.
- ***gcsd2function :*** transformation de nom fonctions et leurs paramètres entrées, sorties et fautes.

IV.5.8 Règles de transformation du GCSD vers le WSMO

Dans cette étape, nous faisons la transformation inverse de ce que nous avons fait précédemment, la figure III.41 montre les transformations inverses du langage de description GCSD vers WSMO.

```
module GCSD2WSMO;
create OUT : wsmo from IN : gcsd;

rule gcsd2servicewsmo {
    from
        s : gcsd!Service
    to
        t : wsmo!Service (NameService <- s.NameOfService

        )
}
rule gcsd2notioncloudwsmo {
    from
        s : gcsd!DeployementModel,
        s1 : gcsd!DeleveryModele,
        s2 : gcsd!Risk,
        s3 : gcsd!Trust,
        s4 : gcsd!Reputation,
        s5 :gcsd!Exprience,
        s6 : gcsd!ConsumerEstimation,
        s7: gcsd!Disponibility,
        s8 : gcsd!MeanTimeToRepaire,
        s9 : gcsd!ReponceTime
    to
        t : wsmo!NotionDeCloud(DeplymentModele <- s.TypeDeplyement, DeleveryModele <- s1.DeleveryModele, Risk <- s2.Risk,
            Trust <- s3.Trust, Reputation <- s4.Reputation, Exprience <- s5.Exprience, ConsumerEstimation <- s6.ConsumerEstimation,
            Disponibility <- s7.Disponibility, MeanTimeToReaire <- s8.MeanTimeToRepaire, ResponseTime <- s9.ReponceTime

        )
}
```

FIGURE IV.41 – Transformation de GCSD vers WSMO

Dans cette transformation, nous utilisons les règles suivantes :

- ***gcsd2servicewsmo :*** transforme le nom de service.
- ***gcsd2notioncloudwsmo :*** transforme les notions de cloud.
- ***gcsd2fundation :*** transformation de fundation.
- ***gcsd2pricing :*** transformation d'éléments pricing.
- ***gcsd2legal :*** transformation des notions legal.
- ***gcsd2interaction :*** transformation de simple et complexe protocole de partie interaction.
- ***gcsd2participant :*** transformation des agents et organisation.
- ***gcsd2sla :*** transforme les notions de SLA SaaS, IaaS et PaaS.
- ***gcsd2ontologyelement :*** transforme le URI de ontologie à partir de ressource de GCSD et le met dans ressource de WSMO dans ontologieelement.
- ***gcsd2transformation :*** transforme de protocole de transport et URI de service.
- ***gcsd2operation :*** transformer le nom de fonction et paramètre d'entrées, sorties et fautes.

IV.6 Conclusion

Dans ce chapitre, nous avons présenté notre solution proposée au problème de et la description de service cloud. Nous avons commencé par introduire la démarche de travail suivie. Puis, nous nous sommes lancées dans la présentation de notre solution en commençant par présenter la description de Cloud. Ensuite, nous avons adapté chaque langage de description « USDL, WSDL, WSMO et OWL-S » au Cloud par l'ajout des notions qui manquent. En dernier, nous avons utilisé la transformation M2M pour transformer chaque langage vers le GCSD proposé et la transformation inverse du GCSD vers les langages de description. Dans le chapitre suivant, nous allons montrer en détail comment nous avons implémenté notre solution en donnant les outils utilisés et le résultat final.

CHAPITRE V

IMPLÉMENTATION

V.1 Introduction

Dans le chapitre précédent de ce mémoire , nous avons proposé une solution de description générique de service cloud. Afin d'illustrer les différentes idées et concepts inclus dans la solution proposée , nous allons l'utiliser pour une expérimentation dans une application java. Pour cela , dans ce chapitre nous allons présenter, dans un premier temps , les outils de développement utilisés , ensuite notre application.

V.2 Outils et environnement de développement

Pour la réalisation de l'application, j'ai utilisé des outils différents qui sont :

- **Java EE** [1] **:** est une spécification pour la plate-forme Java d'Oracle, destinée aux applications d'entreprise. La plate-forme étend Java Platform, Standard Edition (Java SE) en fournissant une API de mapping objet-relationnel, des architectures distribuées et multitiers, et des services web. La plate-forme se fonde principalement sur des composants modulaires exécutés sur un serveur d'applications.
- **Appache TOMCAT** [2] : comme un serveur
  Le serveur Apache Tomcat est un conteneur d'applications Web open source basé sur Java qui a été créé pour exécuter le servlet et les applications Web JavaServer Pages (JSP). Il a été créé dans le cadre du sous-projet Apache-Jakarta. Cependant, en raison de sa popularité, il est désormais hébergé en tant que projet Apache distinct, où il est pris en charge et amélioré par un groupe de bénévoles de la communauté Java open source. Apache Tomcat est très stable et possède toutes les fonctionnalités d'un conteneur d'applications Web commerciales - mais est sous licence Open Source Apache. Tomcat fournit également des fonctionnalités supplémentaires qui en font un excellent choix pour développer une solution d'application Web complète.[40]
- **eclipse-modeling-indigo** [3] : Ce package contient un framework et des outils pour exploiter les modèles : un modélisateur graphique Ecore (diagramme de classe), un utilitaire de génération de code Java pour les applications RCP et le framework EMF, un support de comparaison de modèles, un support pour les schémas XSD, OCL,ATL et les runtimes de modélisation graphique. Il comprend un SDK complet, des outils de développement et du code source.

1. https ://www.eclipse.org/downloads/
2. https ://tomcat.apache.org/download-90.cgi
3. https ://www.eclipse.org/downloads/packages/release/indigo/sr2

- **MySQL Workbench** [4] **: pour la base de données**

  MySQL est un serveur de bases de données relationnelles Open Source. Un serveur de bases de données stocke les données dans des tables séparées plutôt que de tout rassembler dans une seule table. Cela améliore la rapidité et la souplesse de l'ensemble. Les tables sont reliées par des relations définies, qui rendent possible la combinaison de données entre plusieurs tables durant une requête. Le SQL dans "MySQL" signifie "Structured Query Language" : le langage standard pour les traitements de bases de données.
- **protégé** [5] **:** Protégé est un éditeur d'ontologies employé par les développeurs et des experts de domaine pour développer des systèmes basés sur les connaissances (Ontologies) . Protégé n'est pas un outil spécialement dédié à OWL , mais un éditeur hautement extensible , capable de manipuler des formats très divers . Le support d'OWL , comme de nombreux autres formats , sont possible dans protégé grâce à un plugin dédié . Protégé permet aussi de créer ou d'importer des ontologies écrites dans les différents langages d'ontologies tel que : RDF - Schéma , OWL , etc. Cela est rendu possible grâce à l'utilisation de plugins qui sont disponibles en téléchargement [70] .
- **Le modèle MVC :** le MVC signifie Model-View-Controller est un modèle architectural qui sépare une application en trois composants logiques principaux : modèle, vue et le contrôleur. Chacun de ces composants est construit pour gérer des aspects de développement spécifiques d'une application. MVC est l'un des frameworks de développement Web standard les plus fréquemment utilisés dans l'industrie pour créer des projets extensibles et évolutifs.[41]

  -Vue : Une vue est un moyen d'afficher des objets dans une application. Par exemple, l'affichage d'une fenêtre ou des boutons ou d'un texte dans une fenêtre. Il comprend tout ce que l'utilisateur peut voir. La vue est l'interface utilisateur. La vue permet à l'utilisateur d'afficher les données à l'aide d'un modèle et lui permet également de modifier les données.

  -Modèle : Un modèle contient les données utilisées par un programme. Il peut s'agir d'une base de données, d'un fichier ou d'un simple objet. Par exemple, un objet Client récupérera les informations de la base de données, les manipulera et mettra à jour ses données dans la base de données.

  -Contrôleur : Les contrôleurs agissent comme une interface entre le modèle et la vue,

4. https ://www.mysql.com/downloads/
5. https ://protege.stanford.edu/download/protege/4.3/installanywhere/Web*Installers*/

pour traiter toute la logique métier et les requêtes entrantes, manipuler les données à l'aide du composant modèle et interagir avec les vues pour rendre le résultat final. Par exemple, le contrôleur « Client » va traiter toutes les interactions et les entrées de la vue « Client » et mettre à jour la base de données en utilisant le modèle « Client ». Le même contrôleur sera utilisé pour visualiser les données du client.

V.3 Simulation de l'application

Le but de notre application est de créer des services de différents types, ensuite, le consommateur pourra effectuer les opérations suivantes : la recherche, la sélection et l'achat de n'importe quel service comme il le souhaite sans se soucier de son type.
Les conversions d'un langage à un autre se font automatiquement, et le manager peut passer d'un langage à un autre comme il le souhaite.

Dans cette partie, nous allons présenter les principales interfaces de notre application et nous allons essayer de les expliquer avec des exemples.
Notre application est nommée « ChiChiTA ».
Elle comporte 3 acteurs qui sont (administrateur , fournisseur et client)et chaque acteur possède un rôle particulier.

1. **Administrateur**
   - L'administrateur accède à l'application via la saisie d'un mot de passe.
   - L'administrateur est capable de gérer les comptes des clients et des fournisseurs.
   - La gestion des comptes se fait par les actions suivantes :
     (a) **Consulter :** Afficher les informations des deux tables de la base de donné (fournisseur, client).
     (b) **Supprimer :** Suppression d'un compte existant d'un fournisseur ou bien d'un client. Avec un message de confirmation de cette action qui apparait.
   - Vérifier l'état d'ajout de tous les services avec les différents types de description « GCSD, WSDL, USDL, OWL-S et WSMO » :
     — L'affichage d'une table qui contient les informations de tous les services ajoutés par les fournisseurs au marché selon le type de description de service.
     — La communication entre les services : Premièrement, il doit choisir le service qu'il veut transformer. Après, il doit choisir le type d'entrée et sortie (figure IV.1). Le

résultat de entrée apparaît comme le montre la figure IV.2 et sortie comme le montre la figure IV.3.

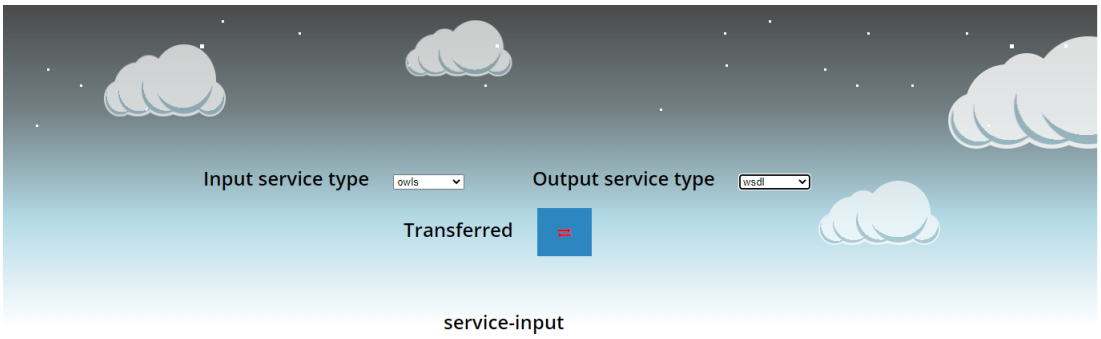

FIGURE V.1 – Interface de choix de type entrée et sortie

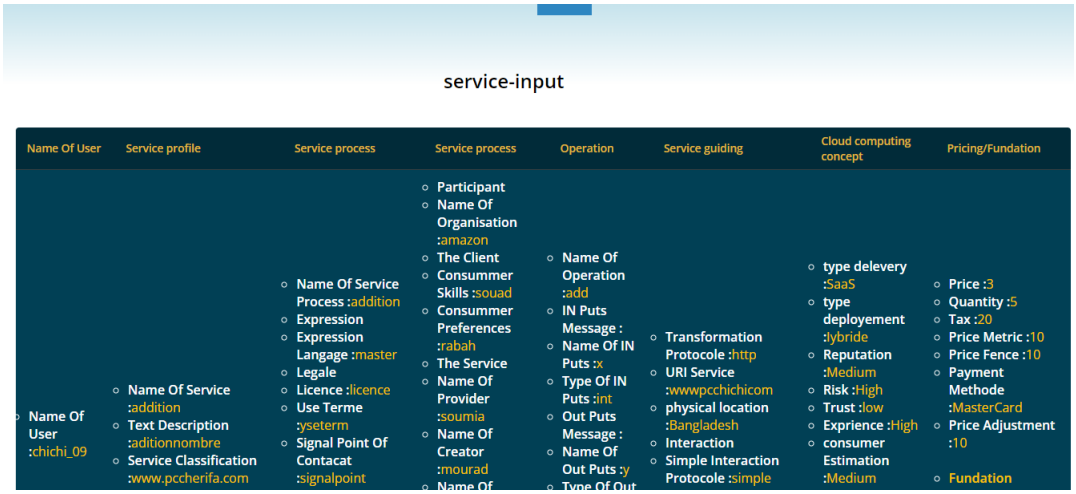

FIGURE V.2 – Résultat de service entrée OWL-S

service-output

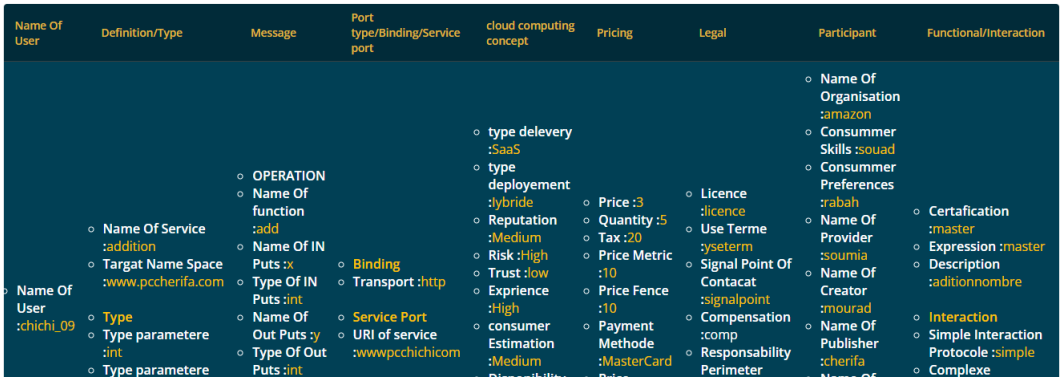

FIGURE V.3 – Résultat de service sortie WSDL

2. **Le fournisseur**
   - **Inscription :** La première fois que le fournisseur accède à l'application, il doit créer un compte, par remplissage d'un formulaire avec ses informations personnelles, telles que (Nom, Prénom, Email, CCP ...), ces derniers seront stockés dans une table de base de données (fournisseur).
   - **Connexion :** Lorsque le fournisseur crée un compte, l'accès à l'application va être facile, il doit seulement saisir son nom d'utilisateur et un mot de passe correct.
   - **Mot de passe oublié :** Si le fournisseur oublie son mot de passe de connexion, il peut le changer par remplissage des informations correctes dans les champs suivants : email, code PIN, le nouveau mot de passe et la confirmation de ce dernier. Le mot de passe sera changé dans la base de données et donne au fournisseur l'accès à l'application.
   - **Modification de profile :** permet au fournisseur de modifier ses informations personnelles.
   - **Ajout des services :** Permet au fournisseur de créer un service selon le type qu'il veut, en entrant les informations nécessaires.

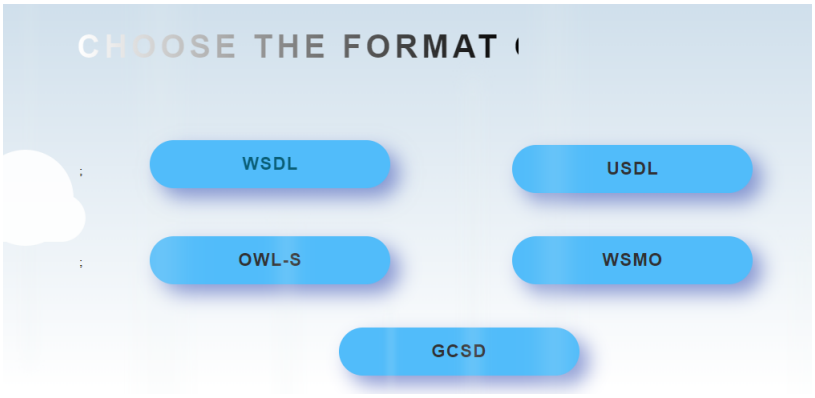

FIGURE V.4 – Choix du type de service à ajouter

On prend un exemple d'ajout de service, par exemple, s'il veut choisir le type GCSD, le formulaire suivant apparait.

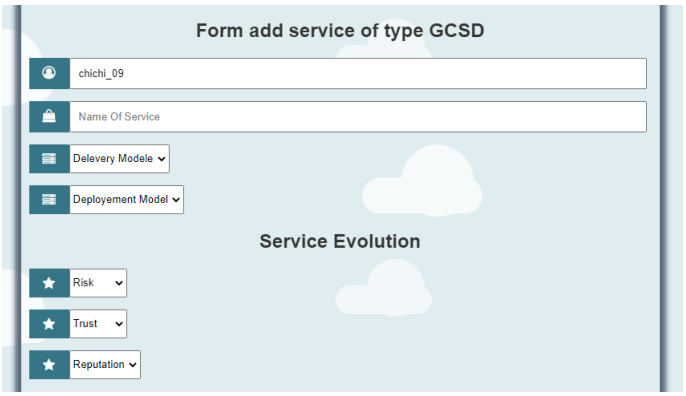

FIGURE V.5 – Formulaire de service GCSD 1

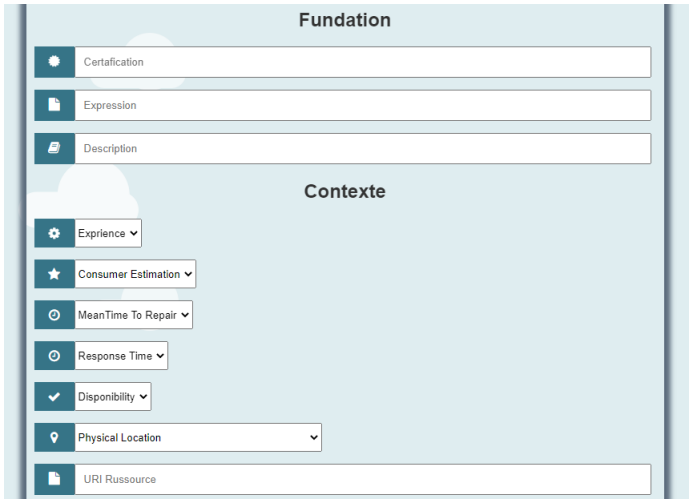

FIGURE V.6 – Formulaire de service GCSD 2

FIGURE V.7 – Formulaire de service GCSD 3

FIGURE V.8 – Formulaire de service GCSD 4

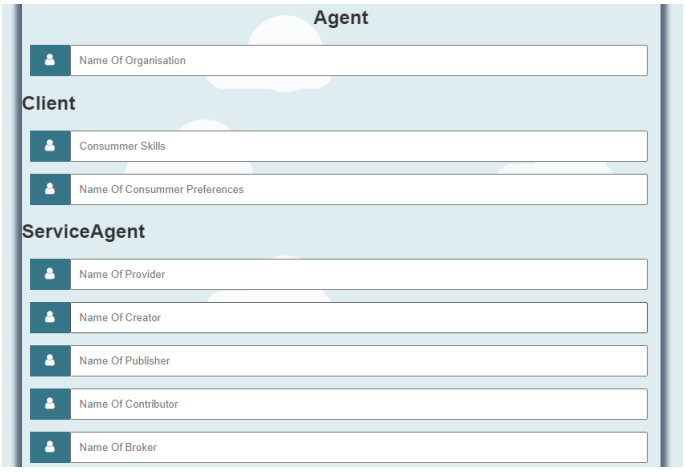

FIGURE V.9 – Formulaire de service GCSD 5

FIGURE V.10 – Formulaire de service GCSD 6

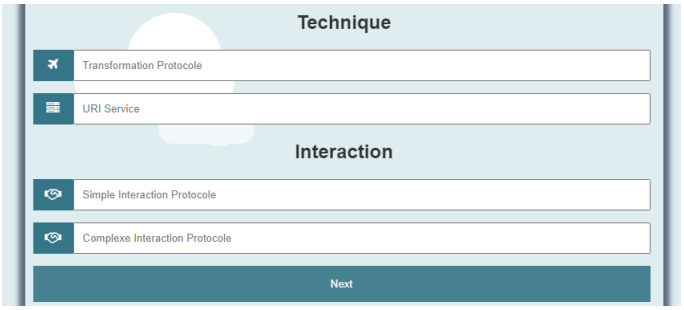

FIGURE V.11 – Formulaire de service GCSD 7

- **Modification de service :** Permet au fournisseur de modifier ses services qui existent déjà.

- **Suppression d'un sevice :** Permet au fournisseur de supprimer un service définitivement de la base de données avec un message de confirmation.

3. **Le client**
   - **Inscription :** La première fois que le client accède à l'application, il doit créer un compte, par remplissage d'un formulaire avec ses informations personnelles, telles que (Nom, Prénom, Email, Carte de crédit...), ces derniers seront stockés dans une table de base de données (client).
   - **Connexion :** Lorsque le client crée un compte, l'accès à l'application va être facile, il doit seulement saisir son nom d'utilisateur et un mot de passe correct.
   - **Mot de passe oublié :** Si le client oublie son mot de passe de connexion, il peut le changer par remplissage des informations correctes dans les champs suivants : email, code PIN, le nouveau mot de passe et la confirmation de ce dernier. Le mot de passe sera changé dans la base de données et donne au client l'accès à l'application.
   - **Modification de profile :** le client peut modifier ses informations personnelles.
   - **Consulter :** Permet au client de consulter les informations des services qui sont proposés par des fournisseurs.
   - **Recherche :** Permet au client de faire la recherche d'un service selon sa contrainte fonctionnelle, si le résultat de recherche =1 alors le client peut faire sa commande sinon si le résultat d'un service>=2 le client peut faire la sélection du meilleur service avec 2 méthodes différentes "MCDM et AHP" selon son choix.

   - **MCDM (Multiple Criteria Decision Analysis)**
     La figure IV.12 représente l'interface de recherche MCDM dans notre application.

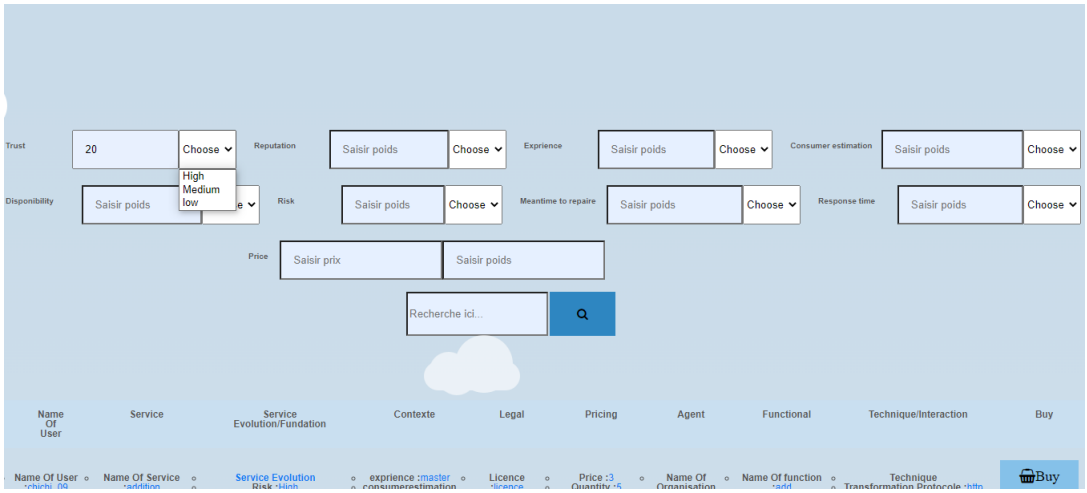

FIGURE V.12 – L'interface de recherche MCDM

MCDM [71] peut être défini comme un ensemble de méthodologies pour la comparaison, le classement et la sélection de plusieurs alternatives ayant plusieurs attributs.

- Etant donné les vecteurs suivants :
  (a) **Services :** Soit *S= S1,S2,S3,...,Si* ensemble des services qui existent dans l'annuaire de broker, tel que $i >= 2$.
  (b) **Critère de performance :** Soit *C= c1,c2,c3,...,Cm* ensemble des critères de service tel que m>=2.
  (c) **Fonctions de mesure de performance :** Soit *F= f1,f2,f3,...,Fm* fi représente la fonction de chaque critère Ci, tel que $m>=2$.
  (d) **Vecteur de description de service de fournisseur :** Soit *D= d1,d2,d3,...,dm* un ensemble qui décrit un service Si, tel que chaque *di* représente la performance d'un service *Si* et *di=fi(si)*.
  (e) **Matrice de décision :** les vecteurs de description de service peuvent être combinés dans une matrice de décision *A* de taille $l * n$ comme suit :

a1,1 a1,2 ... a1,n

a2,1 a2,2 ... a2,n

a1,2 a1,n

Tel que *ai*, est l'évaluation de service *i* et le critère *j* donné par *ai=(Sk)*.

(f) **Vecteur de description de service de consommateur :** Soit *R= r1,r2,r3,...rn* est un vecteur de QoS donné par le consommateur tel que *ri* est la valeur minimale de critère *ci* accepté.

La sélection de service se fait par la comparaison entre le vecteur de QoS de fournisseur *D* et le vecteur de consommateur *R* afin de choisir les meilleurs services.

(g) **Vecteur de poids :** chaque consommateur peut avoir sa propre préférence de QoS, et chaque critère a une importance relative. L'utilisateur exprime la priorité de chaque critère par un poids Tel que *W= w1,w2,w3,. . .,wm*.

- D'abord, nous soustrayons chaque vecteur de la matrice de décision *'A'* la forme vectorielle des besoins de l'utilisateur.
- Ensuite, nous calculons le produit entre le vecteur de poids et résultat de la soustraction de la matrice *'A'*.
- Ainsi, en multipliant la matrice par le scalaire *-1*.
- Puis, nous utilisons une fonction exponentielle pour limiter l'effet de la suppression mutuelle entre les critères supérieurs et inférieurs aux besoins de l'utilisateur.
- Après, on calcule la somme vectorielle de la matrice pour chaque ligne et on la met dans un vecteur list-result.
- En dernier, on fait le tri descendant de list-result pour obtenir la liste des meilleurs services.

  Le pseudo algorithme de la méthode MCDM :

```
Input: S, C, F, D, R, W;
Output: list_result;
Begin
Create matrix A ;
Create float list_result, sum ;
S= {S1,S2,S3,…,Sl} // liste des services
C= {c1,c2,c3,…,cm} // les valeur de critère de fournisseur
R= {r1,r2,r3,…,rm} // les valeurs de critère requièrent
W= {w1,w2,w3,…,wm} // les poids de critère
For (int i=0; i<m; i++ ) { // remplir la matrices
{ For(int k=0; k<l; k++){
A[i][k] = ci− ri ; }}
Sum =0, // la somme d'une ligne de matrice
j=0 ; // indice de la list_result
For (int i=0; i<m; i ++) {
{For (int k=0; k<l; k ++) {
sum= Math.expentiel ((A[i][k] ∗w1)∗(− 1)) + sum;
}
list_result[j] = sum;
j++;
}
Collections.sort(list_result); // trie la liste list_result descendant
Return list_result[0] ; // retourner le Minimum
```

FIGURE V.13 – Pseudo algorithme de la méthode MCDM pour la sélection de service

- **AHP (Analytic Hierarchy Process)**

  La figure IV.13 représente l'interface de recherche AHP dans notre application.

FIGURE V.14 – Sélection du meilleur service selon l'algorithme de AHP

Le principe de l'algorithme AHP [72] est le suivant :

- **Spécifier la matrice suivante :**

La ligne diagonale sera rempli de 1, exemple : le trust par rapport au prix est 9 et le prix par rapport au trust est 1/9. On remplit les restes des champs en suivant cette méthode.

Remarque : On considère que :

1 –> **Importance égale.**

3 –> **Importance modérée.**

5 –> **Forte importance.**

7 –> **Très forte importance.**

9 –> **Extrême importance.**

	Trust	Reputation	Exprience	Consumer estimation	Disponibility	Risk	MeanTimeTorepaire	Response time	Price
Trust	1	5	9	3	5	0,14	7	3	9
Reputation	0,2	1	7	0,2	7	1	0,33	5	0,2
Exprience	0,11	0,14	1	0,11	5	0,14	9	0,2	5
Consumer estimation	0,33	5	9	1	0,33	9	3	0,33	9
Disponibility	0,2	0,14	0,2	3	1	9	5	5	0,14
Risk	7	1	7	0,11	0,11	1	0,14	0,2	7
MeanTimeTorepaire	0,14	3	0,33	0,33	0,2	7	1	3	5
Response time	0,33	0,2	5	3	0,2	5	0,33	1	0,14
Price	0,11	5	0,2	0,11	7	0,14	0,2	7	1

FIGURE V.15 – Matrice de l'algorithme de AHP

- **Calcul du poids :**

  $(1 \times 5 \times 9 \times 3 \times 5 \times 0{,}14 \times 7 \times 3 \times 9)^{1/9} = 2,967786286$

  $(0, 2170, 2710, 3350, 2)^{1/9} = 0,952740102$

  $(0, 110, 1410, 1150, 1490, 25)^{1/9} = 0,603833365$

  $(0, 335910, 33930, 339)^{1/9} = 1,942058419$

  $(0, 20, 140, 2319550, 14)^{1/9} = 0,931732175$

  $(7170, 110, 1110, 140, 27)^{1/9} = 0,787295017$

  $(0, 1430, 330, 330, 27135)^{1/9} = 0,995531859$

  $(0, 330, 2530, 250, 3310, 14)^{1/9} = 0,709809664$

  $(0, 1150, 20, 1170, 140, 271)^{1/9} = 0,634216768$

produit of row	puisance of row
17860,5	2,967786286
0,6468	0,952740102
0,0106722	0,603833365
392,971095	1,942058419
0,5292	0,931732175
0,1162084	0,787295017
0,960498	0,995531859
0,045738	0,709809664
0,0166012	0,634216768
Total of row	10,52500365

- **Calcul de priorité :**

  (2,967786286/10,52500365)=0,281974846

  (0,952740102/10,52500365)=0,090521593

  (0,603833365/10,52500365)=0,057371321

  (1,942058419/10,52500365)=0,184518551

  (0,931732175/10,52500365)=0,088525592

  (0,787295017/10,52500365)=0,074802351

  (0,995531859/10,52500365)=0,067440325

  (0,709809664/10,52500365)=0,067440325

  (0,634216768/10,52500365)=0,060258104

- **Calcul de Lamda max :**

	Trust	Reputation	Exprience	sumer estima	Disponibility	Risk	leanTimeTorepair	Response time	Price	produit of row	puisance of row	priority of row
Trust	1	5	9	3	5	0,14	7	3	9	17860,5	2,967786286	0,281974846
Reputation	0,2	1	7	0,2	7	1	0,33	5	0,2	0,6468	0,952740102	0,090521593
Exprience	0,11	0,14	1	0,11	5	0,14	9	0,2	5	0,0106722	0,603833365	0,057371321
Consumer estimation	0,33	5	9	1	0,33	9	3	0,33	9	392,971095	1,942058419	0,184518551
Disponibility	0,2	0,14	0,2	3	1	9	5	5	0,14	0,5292	0,931732175	0,088525592
Risk	7	1	7	0,11	0,11	1	0,14	0,2	7	0,1162084	0,787295017	0,074802351
MeanTimeTorepaire	0,14	3	0,33	0,33	0,2	7	1	3	5	0,960498	0,995531859	0,067440325
Response time	0,33	0,2	5	3	0,2	5	0,33	1	0,14	0,045738	0,709809664	0,067440325
Price	0,11	5	0,2	0,11	7	0,14	0,2	7	1	0,0166012	0,634216768	0,060258104
Somme	9,42	20,48	38,73	10,86	25,84	32,42	26	24,73	36,48	Total of row	10,52500365	
X=somme of row*priority of row	2,65620305	1,85388223	2,22199127	0,184518551	0,08852559	2,42509222	0,067440325	1,66779923	2,19821565	λmax=somme(x)	13,36366811	

Pour calculer Lamda max

1. On calcule le x :

$$X = somme\ of\ row \times priority\ of\ row$$

2. On calcule Lamda max :

C'est la somme de la colonne x :

2,656203051 + 1,853882235 + 2,221991268 + 0,184518551 + 0,088525592 + 2,425092216

0,067440325 + 1,66779923 + 2,198215646=13,36366811=13,36366811

- **Calcul de Consistency Index(CI) :**

$$CI= (\lambda max - n) / (n - 1)$$

tel que (n) est le nombre de contraintes.

CI= (13,36366811 – 9) / (9 – 1) CI=0,545458514

- **Calcul de Consistency Ratio (CR) :**

Consistency Ratio(CR) = Consistency Index(CI) / Random Index(RI)

CR= CI / RI

CR= 0,545458514 / 1.45

CR=0,376178286

n	Random Index (RI)
1	0.00
2	0.00
3	0.58
4	0.90
5	1.12
6	1.24
7	1.32
8	1.41
9	1.45

Après le calcul de CR, on récupère les services Cloud et on les met dans une matrice suivant cet algorithme :

```
si(i=j) alors
T[i][j]=1
sinon
si (i<j) alors
valeur=cs[i]/cs[j]
si(valeur>0)alors
fonction()   // cette fonction détermine est ce que la valeur est avec une virgule
si oui alors on cherche la valeur rapprochée
T[i][j]=valeur
finsi
sinon
valeur=cs[i]/cs[j]
si(valeur>1)

fonction(valeur)    // cette fonction détermine est ce que la valeur est avec une virgule

si oui alors on cherche la valeur approché
sinon
fonction2(valeur)
finsi
T[i][j]=valeur
finsi
```

FIGURE V.16 – Pseudo algorithme de la méthode AHP pour la sélection de service

- **Acheter :** La dernière fonctionnalité du client qui est la plus importante est d'acheter le service qui le plait après avoir fait les différentes sélections selon son choix (MCDM ou AHP). Tout ce qui reste à faire pour lui est seulement de valider son achat via l'acceptation du SLA qui contient les informations nécessaires sur le service (nom de service , nom de fournisseur, nom de client ...), comme il est capable d'imprimer et télécharger ce SLA.

V.4 Simulation des règles de transformation

- Pour la transformation de langage de service, nous avons utilisé le langage ATL qui est utilisé pour la transformation M2M le projet de transformation contient 3 dossiers "voir la figure IV.17 "

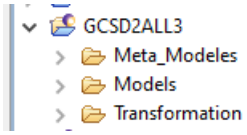

FIGURE V.17 – Les dossiers de transformation

- **Le dossier méta-modèle :** il contient les présentations des descriptions à base de UML. Dans notre transformation, nous avons créé les modèles UML des langages de description WSDL,OWLS, WSMO et USDL, et nous avons créé le langage de description de service cloud GCSD.

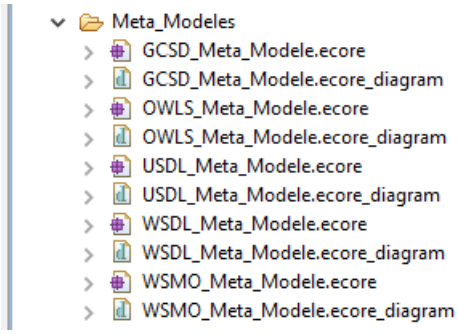

FIGURE V.18 – Le dossier méta-modèle

- **Le dossier Modèle :** cette dossier contient les modèles qu'on veut transformer et les résultats retournés par la transformation. Par exemple, si on veut transformer WSDL vers le GCSD, on écrit comme entrée les notions exprimées dans le méta-modèle de WSDL et il donne le XML de GCSD comme résultat. Si c'est le contraire, on écrit les notions de méta-modèle de GCSD et il donne le XML de WSDL comme résultat.

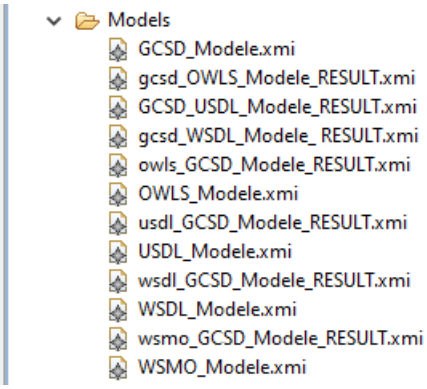

FIGURE V.19 – Le dossier modèle

- Exemple de résultat de transformation vers GCSD.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:gcsd="www.cherifameharezigcsd.com">
  <gcsd:Service NameOfService="test"/>
  <gcsd:DeployementModel TypeDeplyement="public"/>
  <gcsd:DeleveryModele DeleveryModele="saas"/>
  <gcsd:Risk Risk="58"/>
  <gcsd:Reputation Reputation="547"/>
  <gcsd:Trust Trust="14"/>
  <gcsd:Exprience Exprience="66"/>
  <gcsd:ConsumerEstimation ConsumerEstimation="20"/>
  <gcsd:ReponceTime ReponceTime="360"/>
  <gcsd:MeanTimeToRepaire MeanTimeToRepaire="50"/>
  <gcsd:Disponibility Disponibility="80"/>
  <gcsd:Resource URIRussource="www.owls.com"/>
  <gcsd:Pricing Price="25"/>
  <gcsd:PriceAdjustment PriceAdjustment="964"/>
  <gcsd:Quantity Quantity="9524"/>
  <gcsd:Tax Tax="3201"/>
  <gcsd:PriceFence PriceFence="584"/>
  <gcsd:PriceMetric PriceMetric="2154"/>
  <gcsd:PaymentMethode PaymentMethode="visa"/>
  <gcsd:Organisation Name="meharzi"/>
```

FIGURE V.20 – Résultat de transformation vers GCSD

- **Le dossier transformation :** contient les règles de transformation des langages de description de service vers GCSD, et les règles de transformation de GCSD vers les langages de description.

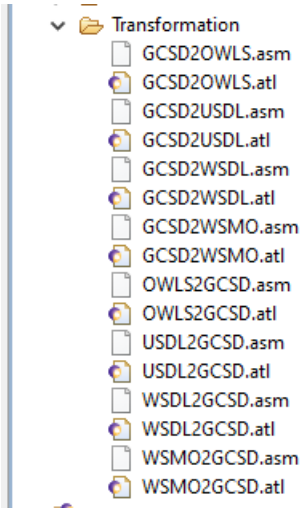

FIGURE V.21 – Le dossier transformation

V.5 Conclusion

Dans ce chapitre , nous avons présenté la mise en oeuvre de notre solution de description des services cloud. Le but d'une telle description est la résolution des problèmes d'interopérabilité qui empêchent la collaboration dans le multi-cloud. Pour tester la cohérence de notre description, nous l'avons testé dans un processus de sélection avec deux algorithmes MCDM et AHP. Aussi, nous avons simulé les règles de transformation pour le mapping entre les différents langages de description.

CONCLUSION GÉNÉRALE

Le cloud computing offrent différentes options (infrastructure , plateforme , logiciel) en tant que services , il permet d'augmenter l'évolutivité des applications et de réduire le coût de l'infrastructure dans l'entreprise . Malgré ces potentiels bénéfiques , les processus de découverte , de sélection et de composition des services cloud posent d'énormes problèmes aux consommateurs , cela est dû au manque de standard pour la description des services cloud. Le but de notre travail est de proposer une solution de description générique et la transformation des langages de description vers cette description .

Nous avons réalisé notre travail en suivant la démarche Extreme Programming. Nous avons commencé par étudier les concepts de base du Cloud Computing et de la description des services. Ensuite, nous nous sommes lancées dans l'étude des travaux existants ayant proposé des solutions pour la description de service cloud et pour la transformation.

Notre solution consiste à établir une description générique des services Cloud et à proposer GCSD, après proposée est un médiateur de transformation, qui joue le rôle de pivot.

Pour illustrer notre proposition, nous avons créé une application Java EE. Notre application permet entre autre la découverte, la création et la sélection des services Cloud. Comme perspective de notre travail, notre description peut être utilisée dans un processus de portabilité ou d'interopérabilité entre plusieurs fournisseurs de services cloud.

Perspectives

Notre projet est ouvert pour ajouter des améliorations comme :

- Ajouter d'autres langages de description adapté au cloud.
- Transformer d'autres langages vers GCSD.
- Utiliser une autre technique pour la transformation.
- Appliquer des ontologies existantes pour dont la découverte de service.

BIBLIOGRAPHIE

[1] P. Mell and T. Grance, "The nist definition of cloud computing," National Institute of Science and Technology, Special Publication, 800, vol. 145, 2011.

Disponible en ligne :

https ://nvlpubs.nist.gov/nistpubs/Legacy/SP/ nistspecialpublication800-145.pdf **[Dernière consultation 24-07-2022]**

[2] Lutz Schubert, Keith Jeffery, and Burkhard Neidecker-Lutz. The Future of Cloud Computing - Opportunities for European Cloud Computing beyond 2010. Technical report, European Commission, 2010.

Disponible en ligne :

https ://www.researchgate.net/publication/281003108 The Future of Cloud Computing. **[Dernière consultation 24-07-2022]**

[3] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, Matei Zaharia. (2010). A View of Cloud Computing .Communications of the ACM , Volume 53, Number 4 (04/2010), Pages 50-58.

Disponible en ligne :

https ://dl.acm.org/doi/pdf/10.1145/1721654 **[Dernière consultation 24-07-2022]**

[4] Furht, B., Escalante, A. : Handbook of Cloud Computing. Springer (2010).

Disponible en ligne :

https ://www.academia.edu/10137405/Handbook of Cloud Computing **[Dernière consultation 25-07-2022]**

[5] D. C. Marinescu, Cloud computing : theory and practice. Morgan Kaufmann, 2017.

Disponible en ligne :

https ://www.elsevier.com/books/cloud-computing/marinescu/ 978-0-12-812810-7 [**Dernière consultation 25-07-2022**]

[6] Philippe Hedde :cloud computing nouveaux modèles, livre blanc, 2012.

Disponible en ligne :

https ://syntec-numerique.fr/sites/default/files/ Documents/20120328-Infra-livreblanc-cloud-computing-nouveaux-modeles.pdf. [**Dernière consultation 26-07-2022**]

[7] Zaid KARTIT.Contribution à la sécurité du Cloud Computing : Application des algorithmes de chiffrement pour sécuriser les données dans le Cloud Storage, thèse de doctorat, Université Mohamed V, Rabat, 2016.

Disponible en ligne :

https ://thesesenafrique.imist.ma/bitstream/handle/ 123456789/1802/THESE KARTIT.pdf ?sequence=1. [**Dernière consultation 26-07-2022**]

[8] LES FONDAMENTAUX DU CLOUD COMPUTING.

Disponible en ligne :

https ://itandsi.files.wordpress.com/2014/08/ livre-blanc-cloud-computing.pdf. [**Dernière consultation 26-07-2022**]

[9] Steve Bennett, Mans Bhuller, Robert Covington « Oracle White Paper in Enterprise Architecture » Architectural Strategies for Cloud Computing(Aout 2009) pp(13-15).

Disponible en ligne :

http ://www.oracle.com/us/ciocentral/ architecrl-strategies-for-cc-396213.pdf [**Dernière consultation 26-07-2022**]

[10] Fang Liu, Jin Tong, Jian Mao, Robert Bohn, John Messina, Lee Badger and Dawn Leaf ,NIST Cloud Computing Reference Architecture Recommendations of the National Institute of Standards and Technology. p. 1.

Disponible en ligne :

https ://nvlpubs.nist.gov/nistpubs/Legacy/SP/ nistspecialpublication500-292.pdf [**Dernière consultation 27-07-2022**]

[11] YAGOUB Mohammed Amine.Une approche basée agent pour la sécurité dans le Cloud Computing, thèse de doctorat, Université Mohamed Khider-Biskra, 2019. **Disponible en**

ligne :

http ://thesis.univ-biskra.dz/4468/1/These.finale.pdf. [**Dernière consultation 27-07-2022**]

[12] R. Buyya, C. Vecchiola, and S. T. Selvi, Mastering cloud computing : foundations and applications programming, Elsevier, 2013.

[13] Etienne Michon, Julien Gossa, Stéphane Genaud, Léo Unbekandt, Vincent Kherbache. Schlouder : A broker for IaaS clouds. Future Generation Computer Systems, Elsevier, 2016, 10.1016/j.future.2016.09.010. hal-01378219

[14] Houssem MEDHIOUB, Architectures et mécanismes de fédération dans les environnements Cloud Computing et Cloud Networking, thèse de doctorat, SORBONNE Université, 2015.

[15] L.F. Noumsl. Etude et mise en place d'une solution "cloud computing " privée dans une entreprise moderne : cas de CAMTEL. Ecole nationale supérieure des postes et télécommunications, 2012.

[16] J-F Pépin, S. Bouteiller, A-S. Boissard, J. Watrinel,Fondamentaux du Cloud computing le point de vue des grandes entreprise.Réseau de Grandes Entreprises (CIGREF).Mars 2013.
Disponible en ligne :
http ://www.eurocloud.fr/doc/cigref2.pdf]. [**Dernière consultation 28-07-2022**]

[17] SAOULI Hamza, Découverte de services web via le Cloud computing à base d'agents mobiles, thèse de doctorat, Université Mohamed Khider-Biskra, 2015.

[18] « what-is-intercloud» .
Disponible en ligne :
https ://www.sdxcentral.com/sdn/network-virtualization/ definitions/what-is-intercloud/
[**Dernière consultation 30-07-2022**]

[19] « what-is-multicloud» .
Disponible en ligne :
https ://avinetworks.com/glossary/multicloud/ [**Dernière consultation 30-07-2022**]

[20] « cloud-interoperability-and-portability ».
Disponible en ligne :
http ://insightaas.com/cloud-interoperability-and-portability-necessary-or-nice-to-have/
[**Dernière consultation 30-07-2022**]

[21]]V. Winkler, "Securing the Cloud : Cloud Computer Security Techniques and Tactics," Syngress 2011

[22] N. Loutas, E. Kamateri, and K. Tarabanis, “A Semantic Interoperability Framework for Cloud Platform as a Service,” in 2011 IEEE Third International Conference on Cloud Computing Technology and Science, (Athens, Greece), pp. 280–287, December 2011.

[23] Trilochan, Anjali Verma, B.Tech Student ”Cloud Computing : Evolution and Challenges”, International Journal of Engineering Science and Computing, Volume 7 Issue No.4, 2017.

[24] J. Zhou , N. Abdullah, and Z. Shi. A hybrid P2P approach to service discovery in the Cloud. In International Journal of Information Technology and Computer Science (IJITCS), MECS, (2011), vol. (3), no1.

[25] Hadjila FethAllah, Composition et interopération des services web sémantiques, thèse de doctorat, université de Tlemcen, 2014.

[26] D.Nicolas BOISSEL-DALLIER ”Réconciliation sémantique des données et des services mis en oeuvre au sein d’une situation collaborative” 2012.
Disponible en ligne :
https ://oatao.univ-toulouse.fr/9220/1/boissel.pdf [**Dernière consultation 09-05-2022**]

[27] Fensel, D., Bussler, C. and Maedche, A. (2002) ‘Semantic web enabled web services’, Sigmod Record ACM, Vol. 31, No. 4, pp. 1-2 **Disponible en ligne :**
https ://link.springer.com/chapter/10.1007/3-540-48005-6-1 [**Dernière consultation 09-05-2022**]

[28] W3C Working Group Note 11 February 2004
Disponible en ligne :
https ://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/ [**Dernière consultation 09-05-2022**]

[29] Turki Hazar ,Leila Baccouche,Henda Ben Ghezala ”ETUDE DE CAS POUR LA SELECTION DES SERVICES WEB BASEE SUR LES CONTRAINTES TEMPORELLES” 2009
Disponible en ligne :
https ://www.researchgate.net/publication/200497263-ETUDE-DE-CAS-POUR-LA-SELECTION-DES-SERVICES-WEB-BASEE-SUR-LES-CONTRAINTES-TEMPORELLES [**Dernière consultation 10-05-2022**]

[30] Booth .D, Haas .H, McCabe .F, Newcomer .E, Champion .M, Ferris .C, Orchard .D, Web Services Architecture (2004),

Disponible en ligne :

https ://www.w3.org/TR/ws-arch/ [**Dernière consultation 10-05-2022**]

[31] Blois .M, Escobar .M, Choren .R, Using agents and ontologies for application development on the semantic web, Journal of the Brazilian Computer Society, 2007.

[32] Extensible Markup Language (XML) 1.0 (Second Edition), https ://www.w3.org/TR/2000/REC-xml-20001006.pdf

[33] J-M. Chauvet, Services Web aves SOAP, WSDL, UDDI, ebXML. . ., Edition EYROLLES, 2022.

[34] G. Gardarin, "Xml des bases de données aux services web, Dunod" 2002.

Disponible en ligne :

https ://side.developpement-durable.gouv.fr/Default/doc/SYRACUSE/82495/xml-des-bases-de-donnees-aux-services-web [**Dernière consultation 10-05-2022**]

[35] Web Services Description Working Group W3C, Web services description language (wsdl)March 2001.

Disponible en ligne :

https ://www.w3.org/TR/2001/NOTE-wsdl-20010315 [**Dernière consultation 10-05-2022**]

[36] Rudi Studer, Stephan Grimm, Andreas Abecker "Web Services : Concepts, Architectures and Applications, Springer"2007.

Disponible en ligne :

https ://link.springer.com/book/10.1007/3-540-70894-4 ?noAccess=true [**Dernière consultation 10-05-2022**]

[37] David L. Martin,Mark H. Burstein "OWL-S : Semantic markup for Web services" 2004.

Disponible en ligne :

https ://www.researchgate.net/publication/39994181-OWL-S-Semantic-markup-for-Web-services [**Dernière consultation 10-05-2022**]

[38] Anupriya Ankolekar , Mark Burstein , Jerry R. Hobbs , Ora Lassila , Drew McDermott , David Martin , Sheila A. McIlraith , Srini Narayanan , Massimo Paolucci , Terry Payne , Katia Sycara, DAML-S : Web Service Description for the Semantic Web, The Semantic Web—ISWC, pages 348–363, 2002.

[39] OWL-S : Semantic Markup for Web Services

Disponible en ligne :

https ://www.w3.org/Submission/OWL-S/ [**Dernière consultation 10-05-2022**]

[40] Aleksa Vukotic, James Goodwill. « Apache Tomcat 7 ».
Disponible en ligne :
https ://link.springer.com/book/10.1007/978-1-4302-3724-2jjn **[Dernière consultation 16-05-2022]**

[41] D. Guamán, S. Delgado and J. Pérez, "Classifying Model-View-Controller Software Applications Using Self-Organizing Maps," in IEEE Access, vol. 9, pp. 45201-45229, 2021, doi : 10.1109/ACCESS.2021.3066348.

[42] Leonard Richardson,Sam Ruby, RESTful Web Services 2007
Disponible en ligne :
https ://www.crummy.com/writing/RESTful-Web-Services/RESTful$_W eb_S ervices.pdf
[Dernière consultation 16-05-2022]

[43] Bouzerzour, N. and Slimani, Y. Towards a MaaS Service for Cloud Service Interoperability. In Proceedings of the 10th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2022), pages 72-83. DOI : 10.5220/0010911400003119

[44] Frédéric Jouault,Freddy Allilaire, The Atlas Transformation Language (Atlas Transformation Language ATL) project Transforming models with ATL.
Disponible en ligne :
https ://www.eclipse.org/atl/documentation/old/ATL$_F lyer_N ormal_V ersion.pdf

[Dernière consultation 17-05-2022]

[45] ATLAS group LINA et INRIA Nantes,ATL :Atlas Transformation Language User Manual - version 0.2 - January 2005.

[46] N. E. H. Bouzerzour, S. Ghazouani and Y. Slimani, "Cloud interoperability based on a generic cloud service description : Mapping OWL-S to GCSD," 2020 IEEE 29th International Conference on Enabling Technologies : Infrastructure for Collaborative Enterprises (WETICE), 2020, pp. 70-75, doi : 10.1109/WETICE49692.2020.00022.

[47] Unified Service Description Language XG Final Report W3C Incubator Group Report 27.
Disponible en ligne :
https ://www.w3.org/2005/Incubator/usdl/XGR-usdl-20111027/L16250
[Dernière consultation 26-05-2022]

[48] T. Mens, K. Czarnecki, and P. V. Gorp, “04101 discussion – A taxonomy of model transformations,” in Language Engineering for Model-Driven Software Development (J. Bezivin and R. Heckel, eds.), no. 04101 in Dagstuhl Seminar Proceedings, (Dagstuhl, Germany), Internationales Begegnungs- und Forschungszentrum fur Informatik (IBFI), Schloss Dag- ¨ stuhl, Germany, 2005.

[49] Anneke KLEPPE, Jos WARMER et Wim BAST : MDA Explained. The Model Driven Architecture : Practice and Promise. Addison-Wesley, 2003.

[50] L.Menet , Formalisation d'une approche d'Ingénierie Dirigée par les Modèles appliquée au domaine de la Gestion des Données de Référence, thèse de doctorat, Université PARIS VIII, 2010.

[51] D. Cetinkaya and A. Verbraeck, ”Metamodeling and model transformations in modeling and simulation,” Proceedings of the 2011 Winter Simulation Conference (WSC), 2011, pp. 3043-3053, doi : 10.1109/WSC.2011.6148005.

[52] Cardoso, J. (2013). Modeling Service Relationships for Service Networks. In : Falcão e Cunha, J., Snene, M., Nóvoa, H. (eds) Exploring Services Science. IESS 2013. Lecture Notes in Business Information Processing, vol 143. Springer, Berlin, Heidelberg. https ://doi.org/10.1007/978-3-642-36356-6$_9$

[53] Jos de Bruijn, Christoph Bussler,John Domingue,Dieter Fensel, Martin Hepp,Uwe Keller, Michael Kifer,Birgitta König-Ries, Jacek Kopecky, Rubén Lara, Holger Lausen ,Eyal Oren, Axel Polleres , Dumitru Roman , James Sciclun , Michael Stollberg. ”Web Service Modeling Ontology (WSMO)” ,3 June 2005. **Disponible en ligne :**
https ://www.w3.org/Submission/WSMO/lang
[Dernière consultation 20-06-2022]

[54] WSMO-Lite : Lightweight Semantic Descriptions for Services on the Web **Disponible en ligne :** *https ://www.w3.org/Submission/2010/SUBM-WSMO-Lite-20100823/*

[55] Salima bourougaa-tria ”Ontologies et Web Sémantique” ,November 2019. **Disponible en ligne :** *https ://www.researchgate.net/publication/337290826*
[Dernière consultation 20-06-2022]

[56] Souad Ghazouani, Yahya Slimani, Towards a standardized cloud service description based on USDL, The Journal of Systems Software (2017), doi : 10.1016/j.jss.2017.06.067

[57] Bu Sung Lee , Shixing Yan , Ding Ma , Guopeng Zhao . Aggregating laas Service . In 2011 Annual SRII Global Conference , San Jose , CA , IEEE , (2011)

[58] Hamdaqa M., Livogiannis T. and Tahvildari L. (2011).A REFERENCE MODEL FOR DEVELOPING CLOUD APPLICATIONS.In Proceedings of the 1st International Conference on Cloud Computing and Services Science, pages 98-103 DOI : 10.5220/0003393800980103

[59] D. Liu and J. Zic . Cloudi : A specification language for modeling Cloud . In 2011 IEEE International Conference on Cloud Computing , Washington , DC , IEEE , (2011) .

[60] P Hoberg , J.Wollerheim , H. Krcmar . Service Description for cloud Service - the customer's perspective . Proceedings of Conlife Academic Conference , Cologne , (2012)

[61] L. Sun , J. Ma and H. Wang . Cloud Service Description Model : An Extension of USDL for Cloud Services . In IEEE Transactions on Services Computing , (2015) .

[62] D. Nguyen , F. Lelli and P. Papazoglou . Blueprinting Approach in Support of Cloud Computing . In Future Internet , Molecular Diversity Preservation International , (2012)

[63] J. Zhou , N. Abdullah , and Z. Shi . A hybrid P2P approach to service discovery in the Cloud . In International Journal of Information Technology and Computer Science (IJITCS) , MECS , (2011)

[64] G. Modica , G. Petralia and O. Tomarchio . Business ontology to enable semantic matchmaking in open Cloud markets . In 8th International Conference on Semantics , Knowledge and Grids (SKG) , Beijing , China , IEEE , (2012)

[65] G. Modica and O. Tomarchio . Matching the business perspectives of providers and customers in future Cloud markets . In Cluster Computing , New York , Springer , (2014)

[66] Gudenkauf, S., Josefiok, M., G¨oring, A., A reference architecture for Cloud service offers. In 2013 17th IEEE International Enterprise Distributed Object Computing Conference (EDOC), September 9-13, 2013, Vancouver, BC, IEEE, pp. 227-236.

[67] Shetty, J., D'Mello, D. A., An XML based data representation model to discover infrastructure services. In 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), May 6-8, 2015, Chennai, India, IEEE, pp. 119-125.

[68] Mastelic, T., Brandic, I., Garcia, A. G., Towards Uniform Management of Cloud Services by applying Model-Driven Development. In 2014 IEEE 38th Annual on Computer Software and Applications Conference (COMPSAC), July 21-25, 2014, Vasteras, Sweden, IEEE, pp. 129-138.

[69] L. Sun, J. Ma, H. Wang, Y. Zhang and J. Yong, "Cloud Service Description Model : An Extension of USDL for Cloud Services," in IEEE Transactions on Services Computing, vol. 11, no. 2, pp. 354-368, 1 March-April 2018, doi : 10.1109/TSC.2015.2474386.

[70] Protégé, *https ://protege.stanford.edu/* **[Dernière consultation 20-06-2022]**

[71] Z. ur Rehman, F. K. Hussain, and O. K. Hussain, "Towards Multi-criteria Cloud Service Selection," in 2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2011, pp. 44–48.

[72] Mingrui Sun, Tianyi Zang, Xiaofei Xu, and Rongjie Wang, "Consumer-Centered Cloud Services Selection Using AHP,"in 2013 International Conference on Service Sciences (ICSS), 2013, pp. 1–6.

[73] F. JoséMoo-Mena, Modélisation des architectures logicielles dynamiques : application à la gestion de la qualité de service des applications à base de services web. Réseaux et télécommunications. Institut National Polytechnique de Toulouse, 2007.

[74] N. Arenaza, Composition semi-automatique de Services Web, mémoire de Master, Ecole Polytechnique Fédérale de Lausanne, 2006.

[75] Pankesh Patel, Ajith H. Ranabahu, Amit P. Sheth, "Service Level Agreement in Cloud Computing" in 2009.

ANNEXE A : REPRÉSENTATION ONTOLOGIQUE DE DESCRIPTION SERVICE

- **Représentation ontologique de description service**

  L'aspect sémantique améliore les résultats de la découverte des services. Il permet une meilleure interprétation et compréhension du sens des concepts liés aux SaaS, PaaS et IaaS.

  Dans notre solution, cette exigence est assurée par l'utilisation d'une ontologie comme support pour notre modèle de description.

  Pour cela , nous utiliserons une ontologie pour décrire les services de cloud computing et leurs caractéristiques sous une forme sémantique . Nous avons utilisé l'outil Protégé pour la conception de notre proposition .

  1. **Représentation ontologique de service :** Notre ontologie représente le module Service qui prend le rôle de la racine, il regroupe tous les notions nécessaires de Cloud computing. Il contient les concepts suivants :

     Technique, deleveryModele, agent, deployementModele, fundation, pricing, serviceEvulition, serviceLevelAgrement, contexe, legal, interaction, functionnal.« Voir les figures 1, 2»

     Chaque concept possède un rôle spécial dans le Cloud. Ces concepts sont détaillés

dans les points suivants :

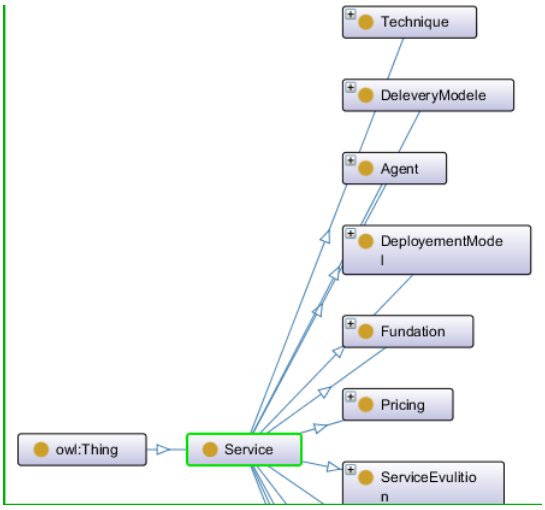

Figure 1 –Représentation ontologique de service part 1

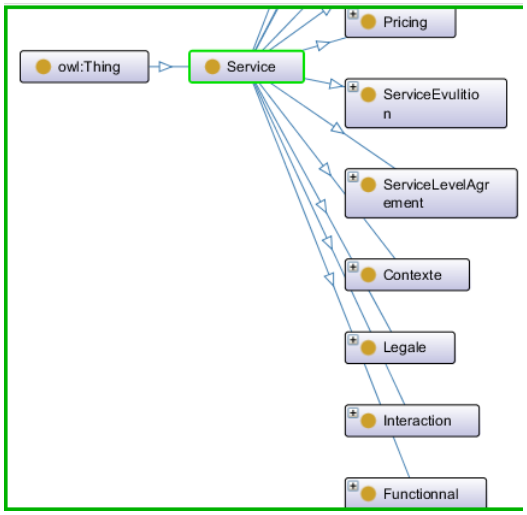

Figure 2 –Représentation ontologique de service part 2

2. **Représentation ontologique de technique et fundation** Dans notre ontologie nous avons ajouté la notion de technique qui est responsable de la transformation de service. Elle regroupe la notion URI qui représente la localisation de service et protocole qui représente le protocole de transformation comme soap et rest.
   La notion fundation fournit un ensemble de propriétés communes telles que le temps, le lieu, etc. Ces propriétés dans notre proposition contiennent 3 notions qui sont « La description, Expression et Certificat ».« Voir la figure 3»

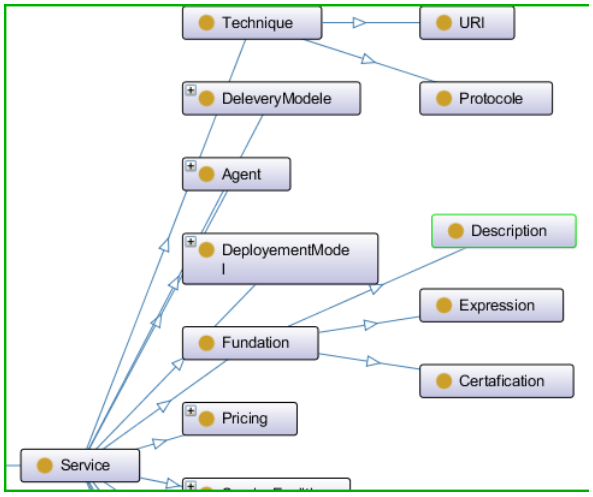

Figure 3 –Représentation ontologique de technique et fundation

3. **Représentation ontologique des Types de services et modèles de déploiement** Nous représentons les types de services par le concept DeleveryModele et ses différents par des sous concepts PaaS, IaaS et SaaS pour représenter les différents modèles de livraisons. Et nous représentons les modèles de déploiement par le concept DeployementModele et ses différents types par des sous concepts communataire, privé et hybride pour représenter les différents modèles de déploiements. La figure 4 représente la partie de types de services et modèles de déploiement sur l'ontologie..

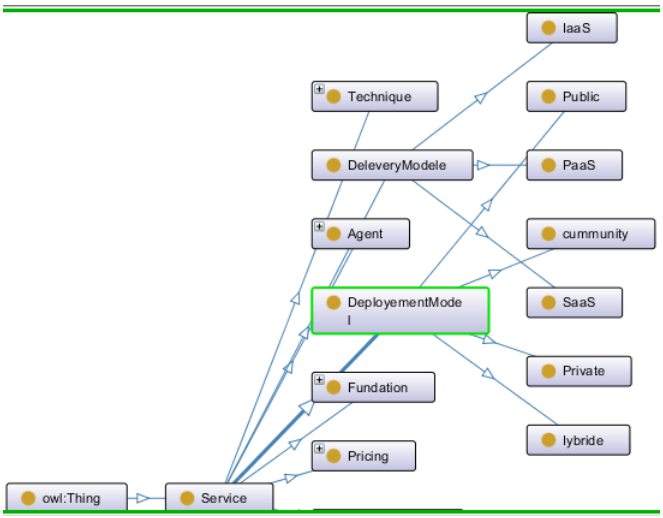

Figure 4 –Représentation ontologique des types de services et modèles de déploiement

4. **Représentation ontologique des agents de cloud computing** Notre ontologie doit décrire les différents acteurs, leurs relations et leurs interactions avec leurs prestations. Donc nous avons classé les agents selon le concept organisation et le concept Person-Role qui représente le rôle de chaque personne. Les personnes sont classés en 2 sous concept, les clients et les services agent. Les clients regroupent « ConsummerSkills et ConsummerPreferences ». Le service agent regroupe les sous concepts suivants « Provider, Creator, Publisher, Contributor, Broker, Carrier et Auditor ».« Voir la figure 5»

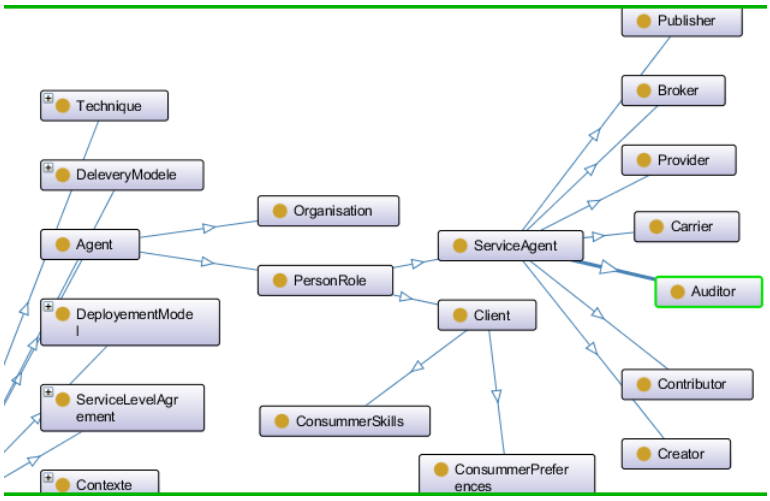

Figure 5 –Représentation ontologique des agents de cloud computing

5. **Représentation ontologique de SLA** Nous présentons un aperçu des éléments d'informations contenus dans les SLA en fonction du modèle de déploiement (SaaS, PaaS ou IaaS). Nous représentons le SLA par le concept ServiceLevelAgrement dans notre ontologie. Le concept ServiceLevelAgrement contient trois sous concepts SLAPaaS, SLAIaaS et SLASaaS. Chaque sous concepts contient des sous concept, commençons par SLAPaaS il regroupe les sous concepts successCode, Incident, ServiceRessource, DisponibilitySLAPaaS, Mangementprotocole et code erro. La figure 6 représente SLA-PaaS.

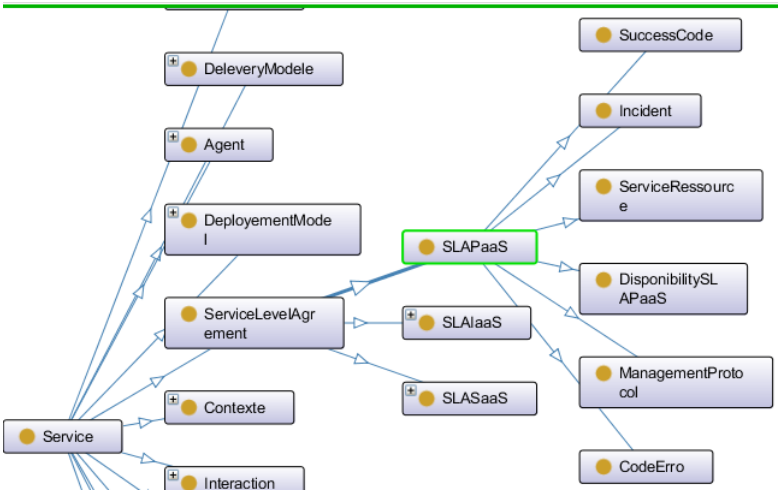

Figure 6 –Représentation ontologique de SLAPaaS

La figure 7 représente SLAIaaS qui regroupe les sous concepts OperationTime, ErrorRate, ServiceTime, IndisponibilityTime.

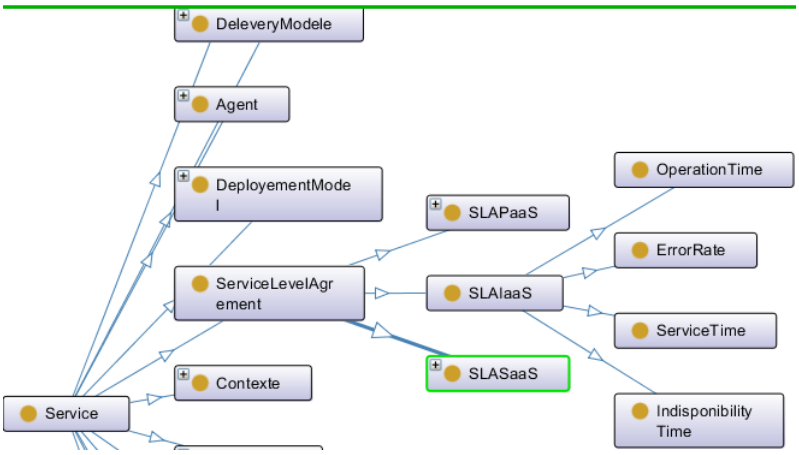

Figure 7 –Représentation ontologique de SLAIaaS

Enfin, la figure 8 représente SLASaaS qui regroupe les sous concepts Reporting, DisponibilitySLASaaS, MeanTimeToRepaireSLASaaS, ResponseTime.

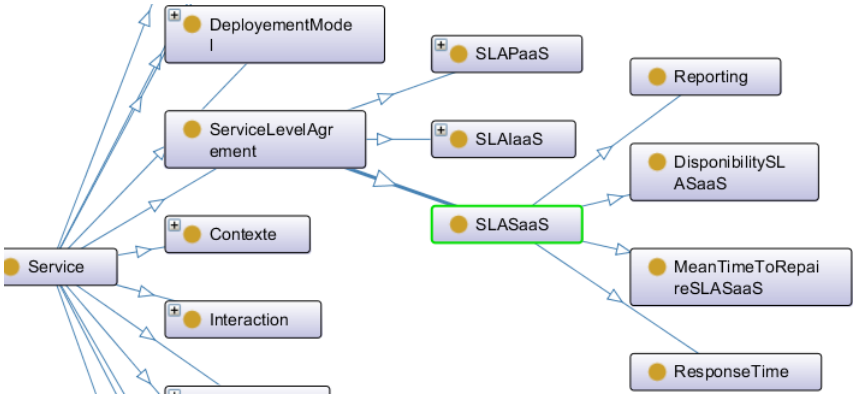

Figure 8 –Représentation ontologique de SLASaaS

6. **Représentation ontologique de contexte** Nous représentons le module contexte par le concept Contexte et ses différentes classes par des sous concepts pour résoudre le manque de notions de cloud computing. La figure 9 est la représentation ontologique du contexte.

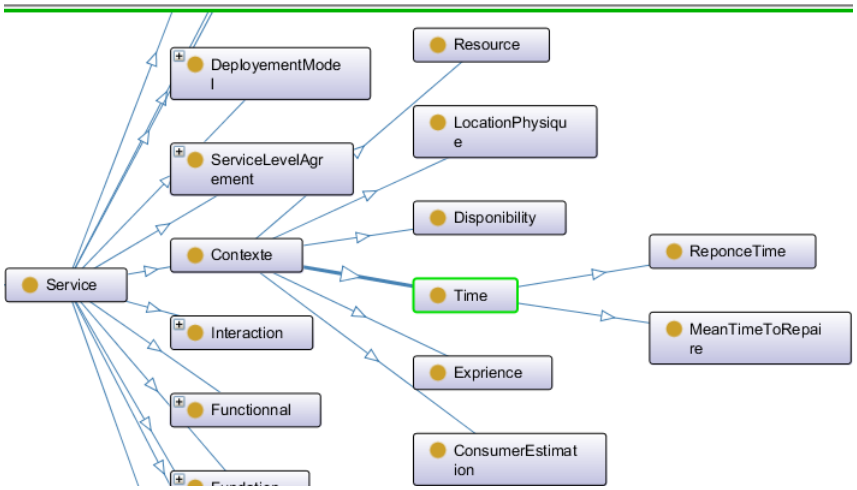

Figure 9 –Représentation ontologique de contexte

7. **Représentation ontologique de Interaction et Functionnal**

   Nous avons représenté le module interaction par le concept Interaction dans notre ontologie et les opérations de service par le concept Functionnal qui regroupe les sous concepts de entrée, sortie et fault. La figure 10 est la représentation ontologique de interaction et opération.

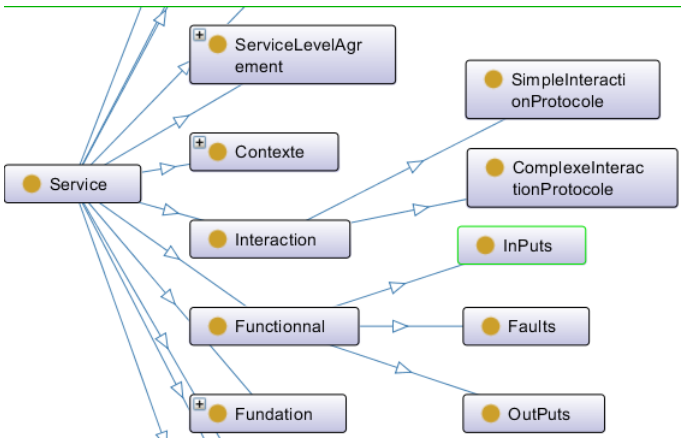

Figure 10 –Représentation ontologique de Interaction et Functionnal

8. **Représentation ontologique de pricing** Nous avons représenté le module prix par le concept Pricing dans notre ontologie et ses différentes classes par les sous concepts . La figure 11 est la représentation ontologique du module Prix :

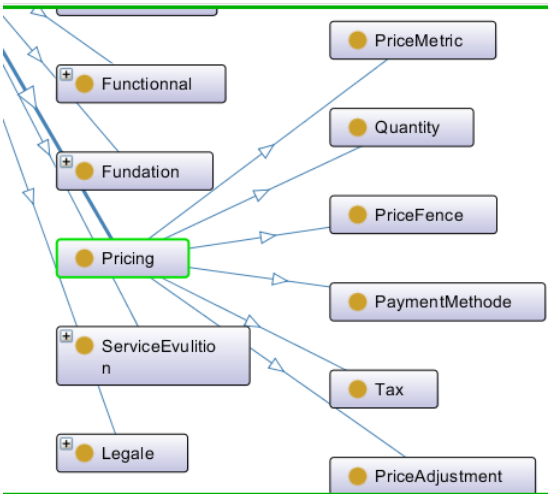

Figure 11 –Représentation ontologique de pricing

9. **Représentation ontologique de ServiceEvaluation** Dans notre ontologie nous avons

rajouté le sous concept ServiceEvaluation qui permet d'évaluer le service cloud à travers les sous concepts réputation , confiance et risque.« Voir la figure 12».

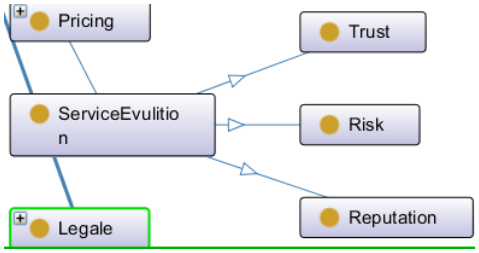

Figure 12 –Représentation ontologique de ServiceEvaluation

10. **Représentation ontologique de Legal** Nous représentons le module legal par le concept Legal dans notre ontologie et ses différentes classes par les sous concepts . Nous avons rajouté le sous concept ResponsabilityPerimeter (pour spécifier les responsabilités) , et SinglePointOfContact pour spécifier un point de contact entre le client et le fournisseur . Nous avons rajouté en plus les sous concepts Compensation (pour exprimer les conditions de dédommagement) , Penalty (pour exprimer les sanctions) , UsedTerms (pour exprimer les conditions d'utilisations) . La figure 13 est la représentation ontologique du module juridique.

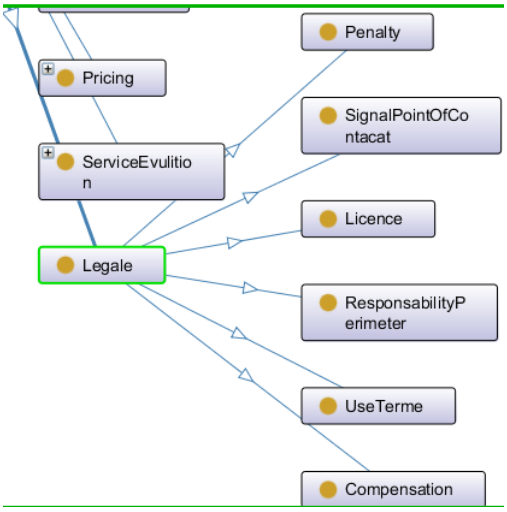

Figure 13 –Représentation ontologique de Legal