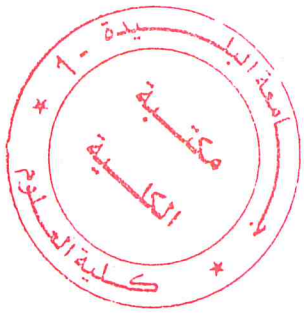
Ministère de l'enseignement supérieur et de la recherche scientifique

Université Saad Dahlab Blida 1



Faculté des sciences

Département d'informatique

Memoire de fin d'étude de master en Systeme d'information et Réseau

Thème

Composition de service Cloud de type Saas

Domaine : MI

Filière : Informatique

Spécialité : Système d'information et réseau

Encadré par :

Mme MANCER Yasmine

Rédigé par :

LAZREG Roufaida
CHITACHANI Ferdous

Présidente
Mme OUKID SALIHA

Année universitaire : 2017/2018

Résumé

Dans l'environnement du Cloud Computing, la composition de systèmes logiciels en tant que service jouent un role trés important pour satisfaire les exigences de l'utilisateur. En général, la composition fait référence à la capacité de communication entre différents services Cloud à travers l'échange des résultats pour atteindre un objectif complexe.

Le présent mémoire introduit une solution pour la composition des services Cloud de type logiciel en tant que service (SaaS). La solution proposée consiste à utiliser l'algorithme Graphplan pour trouver un plan adéquat pour réaliser la communication entre des SaaS et l'algorithme de Djikstra pour optimiser la solution.

Pour l'expérimentation, on a réalisé une application java qui automatise la solution proposée.

Mots clés : Cloud Computing, SaaS, composition, GraphPlan, Djikstra.

Abstract

In the Cloud Computing environment, the composition of software systems as a service plays a very important role in meeting the requirements of the user. In general, the composition refers to the communication capacity between different cloud services through the exchange of results to achieve a complex goal.

This memo introduces a solution for the composition of software-as-a-service (SaaS) Cloud services. The proposed solution is to use the Graphplan algorithm. The goal is to find a suitable plan for communication between SaaS and the Dijkstra algorithm to optimize the solution.

For experimentation, we used a Java application that automates the proposed solution.

Key words **:** Cloud Computing, SaaS, composition, GraphPlan, Djikstra.

الملخص

في بيئة الحوسبة السحابية ، يكون تكوين أنظمة البرمجيات بمثابة نظام يلعب دورًا مهمًا جدًا في تلبية متطلبات المستخدم. بشكل عام ، تشير التركيبة إلى قدرة الاتصال بين الخدمات السحابية المختلفة من خلال تبادل النتائج لتحقيق هدف معقد .

تقدم هذه المذكرة حلاً لتكوين الخدمات السحابية كخدمة (Saas). الحل المقترح هو استخدام خوارزمية غرافبلان لإيجاد خطة مناسبة لتحقيق التواصل بين الخدمات و خوارزمية جيكسترا لتحسين الحل.

لتجربة الحل المقترح, استخدمنا تطبيق جافا يعمل على تنفيذ الحل المقترح تلقائيًا.

الكلمات المفتاحية : الحوسبة السحابية , تكوين ,ساس , غرافبلان , جيكسترا.

Table des matières

LISTE DES FIGURES

Liste des tableaux

Remerciement

Tout d'abord, nous remercions Allah, le tout puissant, qui nous a donné la force, la patience et la volonté pour accomplir ce modeste travail.

Nous tenons à exprimer notre sincère gratitude à notre promotrice ***Yasmine Mancer***, pour son aide capital, et ses conseils qui ont été utiles à la réalisation de ce travail.

Nous remercions les membres du jury qui nous ont fait l'honneur d'accepter de juger notre travail.

Nous voudrons enfin profiter de cette occasion pour dédier notre travail à nos familles avec tous notre amours.

Introduction Générale :

Les solutions de Cloud Computing existantes n'ont pas été construites en tenant compte de l'interopérabilité[1]. Elles verrouillent généralement les clients dans une infrastructure, une plate-forme ou un service Cloud unique empêchant la portabilité des données ou des logiciels créés[2]. En outre, la bataille pour la domination entre les grands fournisseurs, comme Amazon, Google et SalesForce, les rend réticents à se mettre d'accord sur des normes largement acceptées pour promouvoir leurs propres formats incompatibles[3].

Cette domination augmente l'effet de verrouillage et affecte la concurrence puisque les petites et moyennes entreprises (PME) s'opposent à entrer dans le marché Cloud. L'agence européenne pour la sécurité des réseaux et de l'information (ENISA) et la commission européenne ont reconnu le problème de verrouillage des fournisseurs comme un risque élevé que comportent les infrastructures Cloud[4].

La composition est l'élément manquant qui remédiera à cette situation et bénéficiera à la fois aux clients et aux fournisseurs Cloud. En particulier, dans un environnement Cloud interopérable, les clients pourront comparer et choisir des offres parmi les différents offres Cloud de différentes caractéristiques, en négociant avec des fournisseurs Cloud à chaque fois que cela s'avérerait nécessaire, sans mettre en danger les données et les applications et pourront ainsi composer de nouveaux services Cloud à partir de services existants dans le but de satisfaire des demandes plus complexes.

De plus, un marché interchangeable de Cloud ouvrira l'industrie informatique aux PME et renforcera leurs positions sur le marché. Elles inter-opéreront et coopéreront en créant de nouveaux modèles commerciaux selon la demande sans conflits en raison de problèmes d'interopérabilité.

Problématique :

Dans l'environnement du Cloud Computing, la compositionentre les services Cloud est l'un des obstacles qui retarde l'adoption de systèmes logiciels en tant que service. Le problème est que les logiciels Cloud actuels en tant que service n'ont pas été construits au départ avec des mécanismes d'interopérabilité. En général, l'interopérabilité fait référence à la capacité de communication entre fournisseurs Cloud à travers l'échange de services.

Certaines demandes complexes des clients ne peuvent pas être satisfaites par un seul Cloud, nécessitant ainsi la collaboration entre plusieurs Cloud, pour cela une stratégie de composition efficace doit être adoptée.

La problématique de ce sujet consiste à définir unesolutionpour la composition entre Clouds de type SaaS.

Objectifs :

Les objectifs de notre travail sont les suivants :

1. Etude comparative des solutions existantes pour la composition entre Clouds.

2. Définir une solution pour la composition entre Clouds de type SaaS. Cette définition doit prendre en compte les aspects suivants :

   — La stratégie d'ordonnancement des services Clouds qui rentrent dans la collaboration.

   — Description des interfaces de communication des servicesClouds.

   — Description des règles d'interaction entre les servicesClouds.

   — Description des mécanismes de communication jusqu'au résultat final de la composition qui doit être transmis au client.

3. Validation de la solution proposée par une expérimentation en utilisant le simulateur CloudSim.

Organisation du mémoire :

Le présent mémoire est organisé de la manière suivante :

Chapitre 1 : présente un aperçu sur les concepts de base du Cloud Computing, y compris les définitions, les caractéristiques de Cloud, les modèles de service, les modèles de déploiement, les objectifs du Cloud Computing , Inter-Cloud et Multi-Cloud ainsi les obstacles et les avantages du Cloud.

Chapitre 2 : Etat de l'art sur la composition entre Clouds et les solutions existantes pour la composition avec une étude comparative pour faire ressortir les caractéristiques qui rendent encore plus difficile la composition entre Clouds par rapport aux services web.

Chapitre 3 : Modélisation de la solution proposée et ses différents concepts.

Chapitre 4 : Expérimentation de la solution proposée par une application java permettant d'automatiser le processus de composition .

I

Cloud Computing

I.1 Introduction :

Le Cloud Computing est un concept qui regroupe plusieurs technologies permettant à délivrer différents services ; dans lequel les données et les services sont situés sur des endroits à distance qui peuvent atteindre de manière transparente par des appareils(ordinateurs, mobiles, ...) reliées par Internet.

Avec les développements technologiques actuels ont ouvert la voie à l'émergence du concept de Cloud Computing, qui permet l'accès disponible à tout moment sur n'importe quelle machine via l'Internet aux données et les applications logicielles que l'utilisateur veut utiliser , et l'utilisation des logiciels sans avoir installé localement[5, 6].

Dans ce chapitre nous allons présenter certaine définitions proposées par le monde académique. Nous décrivons aussi les caractéristique de Cloud et ses services , sans oublier les types de Cloud et ses objectifs , en mettant l'accent sur l'Inter-Cloud et Multi-Cloud , de plus les avantages et les inconvénients du Cloud Computing.

I.2 Définition de Cloud Computing :

Le Cloud Computing, ou informatique en nuages, est un environnement de stockage et d'exécution flexible et dynamique qui offre à ses utilisateurs des ressources informatiques à la demande via internet. Plusieurs définitions du concept de Cloud Computing ont été proposées :

Le National Institue of Standards and Technology (NIST) défini le cloud computing comme « Cloud Computing est un modèle permettant d'accéder à un accès réseau à la demande à un pool partagé de ressources informatiques configurables (Ex., Réseaux, serveurs, stockage, applications et les services) qui peuvent être rapidement approvisionnés et diffusés avec un effort de gestion minimal ou une interaction avec le fournisseur de services »[7].

Cependant, la définition suggérée par Marston et al[8], est considérée comme la définition la plus complète, basée sur les aspects commerciaux du phénomène de Cloud Computing :

« Le Cloud Computing est un modèle de service technologique de l'information où les services informatiques (matériels et logiciels) sont livrés à la demande aux clients sur un réseau en mode libre-service, indépendamment du périphérique et de l'emplacement. Les ressources nécessaires pour fournir la qualité de service requise sont partagées, évolutives dynamiquement, rapidement provisionnées, virtualisées et libérées avec une interaction minimale avec les fournisseurs de services. Les utilisateurs paient le service comme une dépense d'exploitation sans encourir une dépense d'investissement initiale significative, les services en nuage utilisant

un système de comptage qui divise la ressource informatique en blocs appropriés ».

Selon [9] : « Un Cloud est un type de système parallèle et distribué constitué d'une collection d'ordinateurs interconnectés et virtualisés qui sont dynamiquement provisionnés et présentés comme une ou plusieurs ressources informatiques unifiées basées sur des accords de niveau de service établis par négociation entre le fournisseur de services et les consommateurs ».

D'après toutes ces définitions, nous avons adoptées la définition de NIST.

I.3 Caractéristiques du Cloud Computing :

Pour définir un système Cloud, il faut être basé sur les cinq caractéristiques de bases suivantes[10] :

— Ressources à la demande : Permet de fournir des capacités de calcul informatiques selon les besoins de l'utilisateur (ex : stockage)sans l'intervention humaine.

— Large accès réseau : Les utilisateurs peuvent accéder aux services à l'aide d'un large ensemble de périphériques clients réseau tels que les navigateurs, les téléphones mobiles, tablettes, ordinateurs portables et les postes de travail.

— Mutualisation des ressources : Les ressources sont regroupées pour fournir des services à plusieurs utilisateurs. Ces ressources peuvent être présenter n'importe où et leurs emplacement exact n'est pas consacrées aux consommateurs.

— Élasticité rapide : Selon les besoins de l'utilisateur, les ressources peuvent être libérées et fournies automatiquement.

— Service mesuré : La quantité de ressources consommées est mesurée pour donner la visibilité de l'utilisation pour les fournisseurs et les consommateurs.

Il existe aussi d'autres caractéristiques de points de vue financier, organisationnel et technique[11] :

• Pas d'investissement initial : le Cloud Computing est construit pour satisfaire l'utilisation de ressources à la demande.

• Les couts fixes deviennent variable : plutôt que s'engager à utiliser un nombre donné de ressources pour la durée d'un contrat (habituellement un a trois ans),

le coud permet à la consommation de ressources de changer en temps réel.

• Le grain de l'allocation est fin : le Cloud permet une utilisation minimale en termes de temps et de ressources de changer (par exemple heures d'utilisation de serveurs, octets de stockage).

• Adaptation rapide du nouveau matériel : peut être mis en ligne en quelques minutes pour gérer des modifications imprévues dans la demande, en interne (grosses taches de traitement) ou en externe (trafic vers un site web).

I.4 Modèles de services :

Il existe trois types de services dans le Cloud Computing : SaaS (Software as a Service), PaaS(Platform as a Service), IaaS(Infrastructure as a Service). La (figureI.1) résume ces derniers.

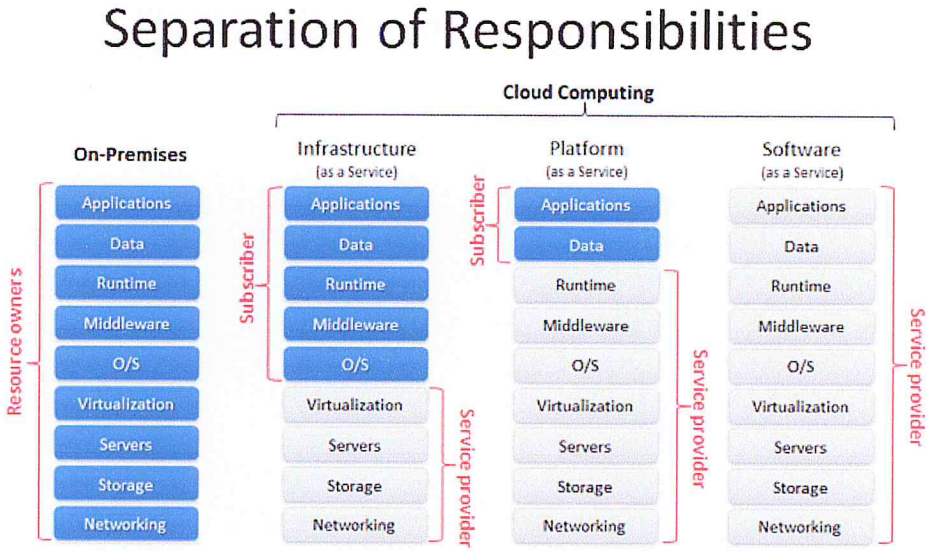


FIGURE I.1 – *S*ervices Cloud[7]

I.4.1 Software as a Service (SaaS) :

Ce modèle de service est caractérisé par l'utilisation d'une application partagée qui fonctionne sur une infrastructure Cloud. L'utilisateur accède à l'application via le réseau et L'administrateur de cette application ne gère et ne suivi pas l'infrastructure sous-jacente (réseaux, serveurs, applications, stockage), et ne contrôle pas les fonctions de l'application[12].

I.4.2 Platform as a Service (PaaS) :

Plateforme as a Service est utilisé pour le développement général des logiciels. Ce modèle permet à l'utilisateur de contrôler les applications et ajouter ses propres outilsfnd[7].

I.4.3 Infrastructure as a Service (IaaS) :

L'Infrastructure as a Service consiste à louer des composants techniques (l'espace de stockage, la bande passante.). Ce service est un standard où une organisation externalise l'équipement utilisé pour Soutenir les opérations, y compris le stockage du matériel, des serveurs, des composants réseau et offre des services à la clientèle sur demande à l'aide d'une interface Web[7].

I.4.4 X as a Services (XaaS) :

La notion XaaS est un terme utilisé par les professionnels de données pour garder une trace de chaque modèle de Cloud Computing, et comment ils se différencient de tous les autres. C'est l'occasion pour les professionnels pour se renseigner et de s'adapter aux nouvelles technologies qui font vraiment la vie plus facile sur le niveau de la gestion des données[13].

Il existe un ensemble des termes XaaS, nous citons quelque unes d'entre elle dans le (tableau I.1) suivant :

Abréviation	Désignation	Définition
DaaS	Desktop as a Service	Le DaaS est un bureau virtuel en nuage. Le fournisseur assure le bon fonctionnement des applications et les mets à jour. Il s'occupe habituellement aussi la sécurité, le stockage et la sauvegarde des données[14].
DaaS	Data as a Service	Des données en tant que service est une stratégie de Cloud utilisée pour faciliter l'accessibilité des données stratégiques de l'entreprise dans un bon moment[15].
FaaS	Framework as a service	Dans les systèmes informatiques, un cadre est souvent une structure en couches indiquant quels types de programmes peut ou devrait être construite et comment ils interagissent[16].
VaaS	Video as a Service	C'est un modèle de distribution hébergé dans le Cloud par un opérateur spécialisé, qui met à la disposition des utilisateurs un réseau sécurisé facturé à l'usage et accessible depuis un navigateur internet[17].
Naas	Network as a Service	C'est un modèle d'affaires pour la prestation de services réseau pratiquement au-dessus de l'Internet et de payer selon l'utilisation ou d'abonnement mensuel[18].
STaaS	STorage as a Service	Ce type est proposé par des services à grand public tels que : AmazonS3, Dropbox, GoogleDrive, etc. Il correspond au stockage de fichiers chez des prestataires externes[19].
WaaS	Workplace as a Service	C'est un environnement de travail virtuelle disponible partout quelle que soit appareil utilisé, il permet un accès sécurisé aux utilisateurs finaux[19].

TABLE I.1 – Abréviation à base de 'XaaS'

I.5 Modèles de déploiement :

Il existe différents types de Cloud Computing présenter dans cette section comme suit :

I.5.1 Cloud Public :

L'utilisateur peut accéder aux ressources n'importe où , ces ressources sont mises à disposition de tout le monde. Cela signifié que les données de l'utilisateur seront particulièrement sécurisées par les serveurs Cloud grâce à l'utilisation de stratégies d'authentification classiques[20].

I.5.2 Cloud Privé :

Dans ce modèle le client est le seul utilisateur du service qui lui est dédié. Le matériel (hardware : les serveurs, dispositifs de copie, pare-feu etc.) peut être opéré par un fournisseur de Cloud. L'accès aux ressources peut être limité au personnel du client, via un réseau local physique ou virtuel (wide area network)[21].

I.5.3 Cloud Hybride :

Ce type de Cloud combine entre le Cloud privé et le Cloud public. Il est utilisé pour réduire au maximum les coûts[20].

I.5.4 Cloud Communautaire :

Dans ce modèle, Un cloud communautaire signifie plusieurs organisations ont les mêmes exigences qu'ils utilisent des ressources partagées[22, 23].l'ensemble des ressources provient de plusieurs organisations ou entreprises qui se les partagent et qui ayant des préoccupations communes (Ex : la mission, les exigences de sécurité, les règles et les considérations de conformité)[12].

I.6 Les éléments constitutifs d'un Cloud Computing :

Selon[24], le CC est composé de :

• **Infrastructure** : L'infrastructure informatique du Cloud est un assemblage de serveurs, d'espaces de stockage et de composants réseau organisés de manière à permettre une croissance incrémentale supérieure à celle que l'on obtient avec les infrastructures classiques. Ces composants doivent être sélectionnés pour leur

capacité à répondre aux exigences d'efficacité et de sécurité.

• **Virtualisation** : est la principale technologie dans le Cloud. Elle consiste à partitionner une ressource physique en plusieurs ressources virtuelles. Par exemple : un serveur, un espace de stockage ou un réseau lors de la création des machines virtuelles. Elle permet d'intégrer les différents serveurs d'une manière plus flexible pour faciliter l'utilisation.

• **Interfaces de service** : placée entre le fournisseur et le client, est un élément de différenciation du Cloud.

• **Centre de données (Datacenter)** : Un centre de traitement de données, en anglais, « Datacenter » est un site physique sur lequel sont regroupés des équipements du système d'information de l'entreprise (mainframes, serveurs, baies de stockage, équipements réseaux et de télécommunications, etc.). Il peut être interne ou externe à l'entreprise, exploité ou non avec le soutien de prestataires.

I.7 Les objectifs du Cloud Computing :

Les objectifs du Cloud Computing ont défini par NIST comme suit[25] :

— Pour illustrer et comprendre les différents services Cloud dans le cadre d'un modèle conceptuel global de l'informatique en nuage.

— Fournir une référence technique à d'autres utilisateurs et consommateurs de comprendre, de discuter, de classer et de comparer les services en nuage.

— faciliter l'analyse des normes internationales pour la sécurité, l'interopérabilité et la portabilité.

I.8 Evolution du Cloud Computing :

I.8.1 Historique de l'évolution du Cloud Computing :

Le concept du Cloud Computing consiste à fournir des ressources informatiques sous forme de services dans lesquels l'utilisateur paie pour ce qu'il utilise.

Ce concept est apparu dans les années 60 notamment avec McCarthy[26], ou encore Kleinrock [27] sous le nom d'informatique utilitaire. C'est ensuite vers la fin des années 90 que ce concept a réellement pris de l'importance avec tout d'abord le Grid Computing [28], ce terme est une métaphore exprimant la similarité avec le réseau électrique dans lequel l'électricité est produite dans de grandes centrales et il est disséminée à travers un réseau jusqu'aux utilisateurs finaux.

Le terme Cloud Computing n'est véritablement apparu qu'au cours des années 2006-2008[29], avec l'apparition d'amazon EC2[30], ou encore la collaboration d'IBM et Google[31, 32], ainsi que l'annonce d'IBM concernant 'Blue Cloud'[33].

D'après une étude menée par Forrester[34], le marché du Cloud Computing s'élevait à environ 5.5 milliards de dollars en 2008, a un peu plus de 23 milliards en 2011 et il devrait atteindre plus de 150 milliards d'ici 2020.

L'évolution du Cloud Computing a donné aussi naissance à de nouveaux concepts tels que le multi-Cloud et l'inter-Cloud.

I.8.2 Inter-Cloud :

L'Inter-Cloud est un réseau global et une extension d'internet "réseau des réseaux". C'est l'interconnexion des infrastructures de plusieurs fournisseurs Cloud.

Une limite majeure du Cloud est que les systèmes Clouds ont peu de ressources physiques. Si un Cloud a épuisé toutes les ressources de calcul et de stockage, il ne peut pas fournir les services à la clientèle. L'environnement d'Inter-Cloud offre différents avantages comme la diversification des lieux géographiques et éviter l'enfermement propriétaire vers le Cloud client. Les avantages pour le fournisseur de services Cloud sont l'exponsion à la demande et des accords de niveau de service pour les clients [35].

I.8.3 Multi-Cloud :

Le terme multi-Cloud est similaire à la fonction du «Cloud-of-Clouds». Le Cloud-of-Clouds c'est l'utilisation de plusieurs services de Cloud Computing dans une seule architecture hétérogène. Ces Cloud peuvent être du même type ou un mélange de types : privés, publics et plusieurs Cloud gérés.

Multi-Cloud est utilisée pour minimiser les risques de perte de données et de temps d'arrêt et d'augmenter la puissance de calcul ou la qualité du service (QoS) et d'élaborer une solution adaptée aux besoins de l'entreprise. Il permet aussi d'améliorer la récupération en cas de désastre, les données sont sauvegardées sur plusieurs Cloud simultanément pour la protection contre les menaces[10].

L'idée de faire appel à de multiples Cloud a été proposée par Bernstein et Celesti[36]. Le rapport du NIST a déclaré que le multi-Cloud peut être utilisé en série lorsqu'une application ou service se déplace d'un Cloud vers un autre ou lorsque les services hébergés sur différents Clouds sont utilisés simultanément[37].

I.9 Avantages et Inconvénients du Cloud Computing :

Le Cloud Computing est comme n'import quel domaine qui a des avantages et des inconvénients cités ces dessous[25] :

I.9.1 Les avantages :

— L'homogénéité : Les systèmes regroupent plusieurs composants identiques ce qui simplifie la gestion et la fiabilité.

— La Virtualisation : La machine virtuelle ne tombe pratiquement jamais en panne ce qui rendre double la fiabilité des systèmes.

— La distribution géographique : Les grandes plates-formes publiques disposent de centres répartis sur la planète pour réduire les risques et placer les données au plus près des utilisateurs.

— Le coût des logiciels est très réduit.

— Les utilisateurs de services n'ont pas besoin de posséder et de gérer les biens d'équipement impliqués.

— En cas de panne sur le lieu de stockage des données, un système de redondance permet le basculement vers un autre site et d'éviter ainsi la rupture de service.

— L'évolutivité et flexibilité : Les entreprises peuvent commencer avec un petit déploiement et croître à un déploiement important assez rapidement, puis les réduire si nécessaire. De plus, la flexibilité du cloud computing permet aux entreprises d'utiliser des ressources supplémentaires aux heures de pointe, leur permettant de satisfaire aux exigences des consommateurs.

— Les utilisateurs ne payent que ce qu'ils consomment et bénéficient d'un haut niveau de service.

I.9.2 Les inconvénients :

Le Cloud Computing présente de nombreux avantages. Cependant, certaines entreprises n'ont pas intérêt à passer à l'informatique dans les nuages, pour des raisons légales et techniques. Parmi les inconvénients que présente le Cloud Computing :

— Problèmes de sécurité et de confidentialité : les données appartenant à l'organisation seront conservées dans un environnement partagé ce dernier est implicitement moins sécurisé qu'un environnement non partagé.

— Interopérabilité : c'est un impact énorme sur l'adoption et l'utilisation du Cloud. Il y a un besoin d'interopérabilité entre les applications du Cloud car d'adoption sera entravée s'il n'existe pas l'intégration des données et des applications à travers les Clouds. Selon certains experts : « Le plus grand défi pour l'adoption à plus long terme des services de Cloud Computing n'est pas la sécurité, mais plutôt l'interopérabilité du cloud et la portabilité des données ».

— Débits réseau importants : Le Cloud nécessite des débits réseau importants pour ne pas décourager les utilisateurs par des temps de réponse trop lent .

I.10 Conclusion :

Au cours des dernières années, le Cloud Computing est considéré comme une technologie émergente qu'a été grandement améliorée[5]. Il est actuellement un concept en plein essor.

Nous avons passé en revue, dans ce chapitre les notions de base relatives à ce concept qui sont les caractéristiques, les trois modèles de services ainsi que les modèles de déploiement qui sont illustrés sous quatre modèles et les objectifs du Cloud, puis nous avons parlé sur la notion d'Inter-Cloud et le Multi-Cloud, les avantages et les inconvénients relatifs à ce concept.

Dans le chapitre suivant, nous présenterons la composition de services Cloud de type Saas.

II

Composition des services Cloud

II.1 Introduction :

Cloud Computing est une collection de ressources accessibles sur le Web en tant que services[38], ces services sont souvent conçus en invoquant plusieurs fournisseurs pour satisfaire les besoins des utilisateurs.

Avec la rapidité de l'évolution des services Cloud, le besoin d'un moteur de composition de services devient une nécessite incontournable. Cette dernière permet l'intégration de diverses ressources de Cloud existant dans un ensemble d'interaction des services pour fournir des solutions répondant à des critères spécifiques[39].

Ce chapitre présente un aperçu des notions de base de la composition des services Cloud.

II.2 La composition des services Cloud :

II.2.1 Définition :

Dans La littérature, plusieurs définitions différentes de la composition des services Cloud sont offerts. Selon les auteurs, la composition des services Cloud est :

1. Une approche évolutive qui consiste à générer des nouveaux services à valeur ajoutée en fusionnant certains services existants afin de fournir un service composite optimal qui inclut des services simples[40].

2. Une phase qui permet de choisir et d'interconnecter les services offerts par différents fournisseurs de services[41], pour augmenter le nombre d'applications de Cloud[42].

II.2.2 Dimensions de composition :

Selon[43] La composition du service Cloud peut être augmentée en deux dimensions :

▷ **La composition horizontale :** Cette dimension consiste la combinaison et l'intégration de services hétérogènes, par exemple stockage, calcul, services de cryptographie, etc.

▷ **la composition verticale :** Cette dimension implique l'intégration de services homogènes, par exemple l'augmentation de la capacité de stockage en ajoutant de nouveaux centres de données.

II.2.3 Étapes de composition :

La composition de services Cloud se déroule en deux étapes[42] :

1. Un schéma de composition est construit pour une demande de composition.

2. La sélection de plan de composition optimal[42]. Cette dernière peut être divisé en deux catégories :

   — La sélection du meilleur service.

   — La sélection d'une composition de services optimale : dans ce type de sélection, un consommateur a généralement besoin de sélectionner un groupe de services composés afin de répondre à ses besoins sur des fonctionnalités complexes[44].

   Le but de la sélection c'est de regrouper et de choisir plusieurs services atomiques ayant des fonctions différentes parmi les services similaires situés sur différents serveurs pour atteindre les Qos les plus élevés en fonction des demandes des utilisateurs[45].

II.2.4 Méthodes de composition :

Il existe trois types de méthodes de composition de services Cloud[46] :

▷ **La méthode de sélection locale :**
Le processus de décision à critères multiples (MCDM) est appliqué pour sélectionner les services concrets appropriés de chaque groupe de services pour la composition. Cette méthode a la meilleure efficacité, alors qu'elle ne peut garantir que des contraintes de QoS locales(par exemple la demande du temps de fonctionnement global minimum du service composite ne peut pas être satisfaite)[47, 48].

▷ **La méthode d'optimisation globale :**
Le problème de composition de service global est transformé en un problème de programmation linéaire en nombres entiers, et résolu par le solveur simple. Il peut rapidement obtenir une résolution de haute qualité, mais il exige que la fonction objective et les contraintes soient des fonctions linéaires [49, 50].

▷ **La méthode d'optimisation intelligente :**
C'est une méthode d'optimisation globale, la différence est que le problème est non linéaire et peut être résolu avec des algorithmes d'optimisation intelligents, tels que l'algorithme Génétiques (GA)[51, 52] et l'algorithme PSO (Particle Swarm Optimization)[53, 54]. Les méthodes d'optimisation intelligentes peuvent résoudre le problème complexe de la composition des services dans un court temps avec une qualité élevée[51, 55], et permet d'obtenir la solution de composition de

service optimale en peu de temps.

◇ **Modalité de composition :**
Selon[43], La composition peut être effectuée selon deux modalités :

1. La composition ponctuelle : les ressources du Cloud sont considérées comme des fonctions, qui reçoivent des exigences en termes de paramètres et retournent la sortie correspondante. Une fois la sortie générée, aucun lien entre les utilisateurs et les fournisseurs n'apparaît.

2. La composition persistante : Cette modalité crée un service virtualisé qui est supposé d'être utilisé par les consommateurs pendant une durée (prédéfinie). Elle est bidimensionnelle peut être augmentée en dimension horizontale et verticale.

II.2.5 Environnements principaux de la composition :

La (figureII.1) présente les environnements de la composition dans le Cloud et les algorithmes appropriés pour chaque un.

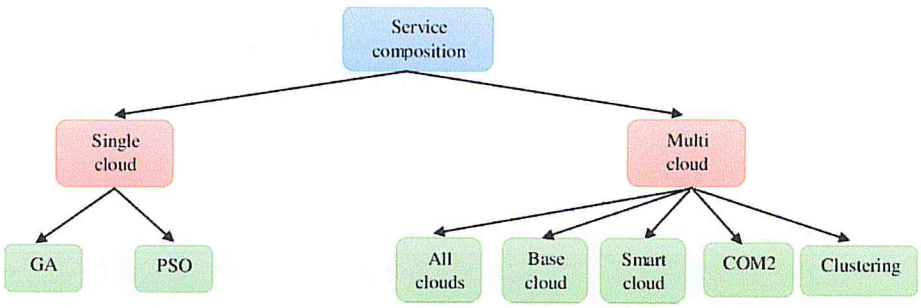


FIGURE II.1 – *Mécanismes de composition de services Cloud*[56]

▷ **Mono-Cloud(centralisé) :**
La stratégie de Cloud unique utilise les services Cloud uniquement car il existe un serveur Cloud unique. Par conséquent, un service parmi SaaS, PaaS et IaaS peut être fourni à tout moment. La gestion du réseau est facile dans cet environnement, mais si un serveur Cloud est défaillant, le système échoue également[56].

Les mécanismes de composition de service dans mono-Cloud se déroule sur deux algorithmes tels que Particle Swarm Optimization (PSO) et Algorithme Génétiqu (AG) :

1. PSO : est une technique d'optimisation basée sur la population. Elle est spécialement conçue pour bien fonctionner dans des environnements dynamiques. IL existe une approche d'ordonnancement basée sur PSO d'apprentissage auto-adaptatif SLPSO son but est de produire un ensemble

de priorités et pour allouer des ressources de Cloud de la manière la plus efficace et aussi d'améliorer la capacité de recherche[51, 52, 53].

2. AG : Les algorithmes génétiques sont des méthodes d'optimisation visant à résoudre des problèmes complexes. Ces algorithmes ont été utilisés pour obtenir l'optimisation globale qui prend en considération l'accord de niveau de service. Cette approche a de bons résultats en cas de temps de calcul, vitesse de convergence, l'évolutivité et la faisabilité, alors qu'il souffre d'un manque de qualité et frais généraux[57].

▷ **Multi-Cloud(distribué) :**
La technique multi-Cloud est l'utilisation simultanée de deux ou plusieurs services Cloud pour diminuer le risque de perte de données[58].

[59] a proposé Les mécanismes de composition de services Cloud dans l'environnement multi-Cloud :

1. Algorithme All Clouds : Cet algorithme considère tous les Cloud comme des entrées pour la composition et énumère toutes les solutions possibles. Cette méthode localise une séquence de composition de service dans un temps d'exécution relativement court.

2. Algorithme Base Cloud : Cet algorithme énumère de façon récursive toutes les possibilités de combinaisons de Cloud jusqu'à ce qu'une solution de composition soit identifiée dans une combinaison. Cette méthode génère une combinaison de Cloud optimale avec un petit nombre de Cloud ; cependant, cela nécessite un temps d'exécution important.

3. Algorithme Smart Cloud : Cet algorithme conçu pour trouver une combinaison de Cloud quasi-optimale après la recherche d'une séquence de composition de service à un coût réduit mais il prend beaucoup de temps.

4. Algorithme COM2 : L'algorithme COM2 permet de choisir le Cloud avec le nombre maximal de services, ce qui accroît la possibilité d'exécution de demandes de service avec un minimum de frais généraux[42].

5. Clustering : Cet algorithme fourni un ensemble de services Web unique avec une capacité de combinaison afin d'obtenir les exigences complexes des utilisateurs[56].

▷ **Étude comparatives :**
Les algorithmes qui sont déjà cités ci dessus sont comparés selon[56], l'algorithme base Cloud et clustering Cloud est certainement mieux que d'autres algorithmes dans la recherche de la composition optimale de Cloud. En outre, COM2

et algorithme All Cloud sont certainement mieux que d'autres algorithmes en terme de temps de marche. Il y avait un problème courant parmi les méthodes existantes, c'est l'équilibrage de la charge entre les serveurs Cloud.

Le (tableauII.1) ci-dessous contient une comparaison sur les mécanismes cités en[56] :

Algorithmes	Avantages	Inconvénients
All Cloud	Temps d'implémentation rapide.	Échec à trouver la composition optimale du Cloud.
Base Cloud	Trouver la composition optimale dans tous les cas. ensemble réduit de Cloud.	Le temps de fonctionnement est très élevé. Équilibrage de charge inefficace.
Smart Cloud	Fournir une combinaison de Cloud presque optimale. Un ensemble réduit de Cloud.	Le temps est long. Équilibrage de charge inefficace.
COM2	Trouver la composition optimale dans certains cas. un court temps d'exécution.	Équilibrage de charge inefficace.
Clustering	Trouver la composition optimale du Cloud dans tous les cas. Haute capacité de combinaison.	Équilibrage de charge inefficace.

TABLE II.1 – Comparaison d'algorithmes

II.3 Classification des méthodes de composition :

Dans la littérature, les méthodes peuvent être divisées en 3 catégories distinctes, y compris basée sur le cadre, basé sur l'agent et basé sur l'heuristique[41].

II.3.1 Solution à base de framework :

Les mécanismes basés sur le cadre sont basés sur un ensemble d'hypothèses, de concepts, de valeurs et de pratiques qui constituent et forment un moyen de voir la réalité. Elle sont utilisées afin d'organiser et de gérer pour la recherche, la sélection et la composition des services de Cloud Computing par des approches complètement nouvelles[41].

La classification Framework comprend quatre volets principaux[42] :

1. Un environnement multi Cloud (MCE) est un ensemble de Cloud. Chaque Cloud contient un ensemble de fichiers de service et chaque service fichier contient un ensemble de services.

2. L'interface utilisateur accepte une demande de l'utilisateur et affiche la séquence de composition de services.

3. Le mélangeur de Cloud sélectionne la série de Cloud appropriés (la série qui a le plus de services adaptés pour répondre à lademande de l'utilisateur) et génère une liste de combinaison de Cloud en fonction de l'ensemble.

4. Le compositeur reçoit le service Cloud résultant de la liste combinée et détermine les services dans chaque Cloud. En fonction de la sélection, le compositeur service produit la séquence de composition de services.

La méthode Framework-based est l'une de classifications utilisée par plusieurs auteurs. Les travaux suivants présentent l'utilisation de cette dernière.

Selon G. FAN et AL[41], ont proposé une méthode pour filtrer les attributs de l'élément de base de la composition de services pour la description de leur relation.

le modèle proposé comprennent :

1. Un réseau fiable de composition de service est défini pour décrire les différents éléments de la composition de services.

2. Une stratégie de composition fiable et son algorithme d'application sont utilisés pour allouer dynamiquement le service disponible.

3. La sémantique opérationnelle et les théories de Petri nets aider à prouver l'efficacité et l'efficience de stratégies de composition fiables. Par conséquent, ce cadre présente des avantages majeurs tels que l'augmentation de l'évolutivité et de l'efficacité, alors qu'il ne peut être utilisé que pour les services SaaS.

Selon JULA et AL[60], les fournisseurs de Cloud doivent fournir un compositeur de service composite (CSC), qui est un ensemble de composants qui recherche la meilleure composition des services uniques pré-fournis en fonction des besoins des clients. En fonction d'optimisation du temps de service dans le Cloud Computing Service Composition (STOCCSC), un algorithme de concurrence impérialiste (CSSICA) est proposé. CSSICA est capable de réduire le temps de service composite fourni.

KURDI et AL[42] ont proposé un algorithme d'optimisation combinatoire pour la composition de services de Cloud qui peuvent utiliser plusieurs Cloud. Cette algorithme s'assure que le Cloud avec le nombre maximal de services sera toujours choisi avant d'autres, ce qui accroît la possibilité d'exécution de demandes de service avec un minimum de frais généraux, la réduction du temps est une grande efficacité qui est présenté par ce mécanisme mais il souffre du cout élevé.

X.WANG et AL[61], ont proposé un modèle de sélection de service Cloud qui adopte les courtiers de services Cloud et une stratégie de sélection de service dynamique appelée DCS. Au cours du processus de sélection des services, chaque courtier de services Cloud gère certains services de Cloud en cluster et exécute la stratégie DCS avec un mécanisme d'apprentissage adaptatif. Ce dernier est conçu pour optimiser dynamiquement la sélection du service Cloud et renvoyer le résultat de service le plus optimal à l'utilisateur. Cette stratégie a une meilleure efficacité globale dans l'acquisition de solutions de service de haute qualité à un coût de calcul inférieur, mais elle souffre de temps.

II.3.2 Solution à base d'agent :

Selon[62], une approche basée sur l'agent est proposée pour composer des services dans des environnements multi- cloud pour différents types de services Cloud.

Les agents sont dotés d'un protocole net Semi-récursif et des tables de capacités de services pour composer des services en fonction des besoins d'utilisateurs[41].

1. Le protocole net Semi-récursif : Ce protocole est basée sur des appels récursifs .Les agents dans ce protocole ont deux rôles : initiateur et participant. Un consommateur adoptant le rôle d'initiateur diffuse un appel pour réaliser une tâche à n participants (entrepreneurs). Les participants peuvent répondre soit avec : une proposition (citation) pour exécuter la tâche ou par un message de refus. A partir des propositions reçues, l'initiateur sélectionne la meilleure proposition (la moins chère) et envoie : une

proposition d'acceptation au plus grand participant, et des messages de rejet aux autres entrepreneurs restants. Après l'exécution de la tâche, le participant sélectionné envoie soit : un message de résultat d'information, soit un message d'échec en cas de résultats infructueux[41].

2. Tables de capacités de service (SCT) : Ces tables sont utilisés par les agents pour enregistrer et consulter des informations sur les participants Cloud et leurs statuts (service occupé, service capable d'exécuter une tache a un moment donné)[41]].

Les SCT sont[41] :

— dynamiques : car les agents peuvent être supprimés ou ajoutés.

— précis : sont exact car les adresses et les fonctionnalités des agents sont correctes.

— incomplètes : car les agents peuvent ne pas connaitre la liste complète des agents existants

La méthode basé sur l'agent a une grande efficacité et d'évolutivité, tandis que l'inconvénient de cette approche est un haut moment de la composition de services.

▷ **Architecture de composition de service Cloud basée sur l'agent :**
La (figureII.2) montre les six(6) composants de la méthode basé sur l'agent[41] :

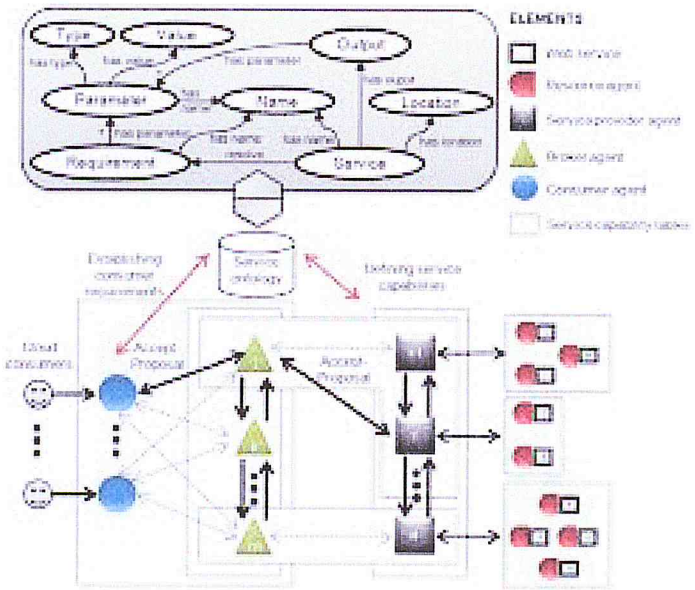


FIGURE II.2 – *Architecture de composition de services à base d'agent*[41]

— Ontologie de service : Ce composant fournit la spécification de service qui décrit la fonctionnalité, l'entrée et la sortie des services.

— Services Web : Ce sont des interfaces avec des logiciels ou des ressources (Cloud) accessibles à distance.

— Agents de ressources (RA) : Ces éléments reçoivent des demandes pour résoudre les exigences des fournisseurs de service et traite ces demandes via leur services web associé, en renvoyant la sortie au fournisseurs.

— Agents de fournisseur de services (SPA) : Ils gèrent les ressources des fournisseurs de Cloud en contrôlant et en organisant les RA.

— Agents de courtage(broker) (BA) : Ces composants composent et fournissent un seul service virtualisé aux clients du Cloud.

— Agents de consommateur (CA).

Un algorithme de composition de service automatisé basé sur un agent a également été proposé par A. SINGH et Al, comprenant des phases de traitement des demandes et de composition automatisée des services, non seulement responsable de la recherche de services complets, mais aussi la réduction du coût des machines virtuelles consommées uniquement par les services à la demande.

Dans cette étude, une stratégie d'affectation intelligente et automatisée a été présentée pour affecter des ressources dans un environnement Cloud Computing. Dans ce mécanisme, plusieurs agents intelligents ont été déployés pour réduire la complexité du système . L'agent de courtage facilite la recherche du centre de données optimal pour les besoins de chaque utilisateur et la composition du service au nom de l'utilisateur. Ainsi, le mécanisme proposé permet à l'utilisateur de trouver un fournisseur de services optimal dans n'importe quelle condition et assure une allocation de service efficace dans les centres de données. Cependant, il souffre d'un temps d'exécution élevé[41].

BASTIA eT AL, ont proposé une approche multi-agents dans des environnements multi-Cloud pour différents types de services Cloud. Dans cette étude, les participants au Cloud et les ressources sont implémentés et instanciés par des agents. Ils ont pu réduire le nombre de messages de passage de moitié afin d'augmenter l'efficacité globale. En outre, ils ont planifié un auto-organisation (MAS) à 2 niveaux (3 niveaux de multi-agents) de sorte qu'une composition de service de Cloud soit établie. Les résultats obtenus ont révélé que ce mécanisme a une efficacité et une évolutivité élevées, mais il souffre d'un sur coût élevé pour les agents[41].

II.3.3 Solution à base d'heuristique :

QIANG YU ET AL, ont défini et étudié le problème de composition de service Web dans un environnement multi-cloud. Ils ont proposé deux algorithmes de combinaison de Cloud variés :

1. L'algorithme Greedy-WSC choisit le Cloud qui peut fournir les services les plus demandés jusqu'à ce que les Clouds sélectionnés couvrent toutes les exigences.

2. L'algorithme ACO-WSC, les fourmis artificielles voyagent afin de construire des combinaisons de Clouds. La colonie de fourmis obtient itérativement la solution optimale.

La méthode proposée permet de trouver efficacement des plans de composition de services avec un nombre minimal de Clouds. Cependant, ces approches souffrent d'une grande complexité[41].

KHOLIDY et AL[63], ont discuté du problème de la composition du service Cloud en tant qu'optimisation multi-objectif afin que les exigences de QoS de l'utilisateur soient satisfaites, et ont présenté une approche pour résoudre le problème multi-objectif en le modifiant en un seul objectif. Cette approche a utilisé l'algorithme génétique pour résoudre le problème de composition. Ensuite, le système de composition décide quels fournisseurs SaaS et IaaS doivent être sélectionnés pour l'utilisateur final. L'objectif final est trouvé une composition de service Cloud pour minimiser le coût et le temps et améliorer le débit[41].

Karimi et AL[64], ont identifié les exigences de la composition du service en fonction du contrat SLA dans les environnements de cloud computing. Puisque les services ont été sensiblement augmentés dans un environnement cloud et que les clients se préoccupent de la rapidité de la prestation de service et de la qualité de service, les chercheurs ont utilisé des techniques de data mining telles que le clustering, les règles d'association l'espace de recherche d'un problème. Selon les résultats de l'étude, il est clair que ces techniques peuvent entraîner la réduction du temps de composition du service et l'amélioration de l'optimalité des services composés. De plus, les résultats de l'étude indiquent que la méthode proposée est évolutive ; par conséquent, il est hautement approprié pour les environnements de cloud dynamique. Mais il souffre de frais généraux élevés[41].

Selon[46] un algorithme de colonie d'abeilles artificielles (ABC) est proposé par Karaboga, qui est appliquer au problème de composition de service et utilise l'exploration des abeilles pour simuler la recherche de la solution de composition optimale du service.

C'est un algorithme de recherche aléatoire heuristique. Dans cet algorithme, chaque source de nourriture représente une solution réalisable au problème à

résoudre, et la qualité de nectar de la source de nourriture représente l'aptitude de cette solution réalisable. D'après cet algorithme Les abeilles sont classées en trois groupes pour obtenir une solution optimale ou quasi-optimale :

— Abeilles employées : Ce sont responsables d'exploiter le voisinage de la source de nourriture et de partager leurs informations avec les abeilles qui attendent dans la ruche.

— Les spectateurs : Ils attendent des informations dans la ruche, choisissent une source de nourriture et l'exploitent.

— Scouts : Quand aucune information n'est mise à jour après plusieurs itérations, l'abeille employée deviendra une éclaireuse et trouvera aléatoirement une nouvelle source de nourriture pour commencer une nouvelle recherche.

II.4 Comparaison des méthodes de composition :

Le (tableau II.2) met l'accent sur des critères de comparaison entre les différents travaux déjà cités.

Classification	Référence	Type de service	Sélection
Framework	[41]	SaaS	Oui
	[60]	Non mentionné	Non
	[42]	Non mentionné	Oui
	[61]	Non mentionné	Oui
Basé sur l'agent	[62]	Tous types de services	Oui
	[41]	IaaS	Non
	[41]	Tous types de srvices	Non
A base heuristiques	[41]	Non mentionné	Oui
	[63]	SaaS et IaaS	Oui
	[64]	Non mentionné	Non mentionné
	[46]	SaaS	Oui

TABLE II.2 – Étude comparative des méthodes de composition

II.5 Discussion :

Dans les sections précédentes, nous avons décrit les méthodes de composition de services de Cloud dans trois catégories principales : basées sur un framework, basées sur des agents et basées sur des heuristiques.

Quatre solutions parmi sept(environ 36%) ont utilisé la méthode basé sur un framework, elle offre une grande flexibilité pour composer les services de Cloud Computing et a un temps de composition faible. Cet mécanisme convient donc à tout environnement Cloud en tant que Cloud unique et multicloud. En outre la méthode basé sur l'agent est traités dans trois solutions parmi sept(environ 27%) cité au dessus. Cette méthode est également adapté à l'environnement multi-cloud car il est de haute qualité pour la gestion des tâches mais elle souffre d'un temps de composition élevé. Ainsi la méthode heuristique(environ 36%) est adapté à un environnement de Cloud unique et se concentre sur l'optimisation et l'efficacité de la composition mais le temps de composition est élevé.

II.6 Conclusion :

La composition des services Cloud, l'un des problèmes les plus importants dans le domaine Cloud Computing , a incité les chercheurs à proposer de nouvelles techniques et méthodes pour trouver des solutions optimales en moins de temps.

Dans ce chapitre, nous avons présenté la composition de services Cloud en passant par : la définition, les dimensions, les étapes et les méthodes de composition ainsi les environnements principaux de la composition. Nous avons fini par une classification et une comparaison des solutions existantes de la composition de services Cloud.

Nous avons conclus que la plupart des solutions existantes sont situées dans la méthode basée sur framework et basée sur l'heuristique.

Dans le chapitre suivant, nous présenterons un modèle de composition pour les logiciels en tant que service(Saas) dans l'environnement du Cloud Computing.

III

Modélisation de la composition des services Cloud

III.1 Introduction :

Après avoir effectué une étude comparative des solutions existantes pour la composition des services Cloud dans le chapitre précédent, nous allons dans ce chapitre, présenter notre solution pour la composition des services de type SaaS, l'objectif de ce chapitre est de concevoir un processus qui pourra faire la composition entre les différents SaaS.

Notre processus commence par la description des différents Saas dans des fichiers WSDL(Web Services Description Langauge), ainsi par des règles d'interaction semi-formelles qui seront traduits par la suite en langage de prédicats pour que le planificateur Graphplan puisse l'utiliser. A la fin, et Après avoir décrit les différents SaaS et trouver le chemin pour la collaboration,la dernière étape consiste à effectuer la composition entre les différents SaaS en utilisant le langage BPEL(Business Process Execution Language).

Démarche de travail :

Le travail que nous avons fait afin de réaliser notre projet est suivi par la méthode XP-eXtreme Programming[65], qui représente un processus de développement Agile, dont les ressources sont régulièrement actualisées.

Cette méthode est conçue pour améliorer la qualité et la capacité du logiciel à s'adapter correctement aux exigences changeants du client[66].Elle a été conçue à l'origine par Kent Beck et Ron Jeffries pendant le projet "C3" (Chrysler Comprehensive Compensation System)[67].

Extreme Programming met l'accent sur un travail d'équipe collaborative, et met en œuvre un environnement simple et efficace permettant aux équipes de devenir hautement productives[68].

L'Extreme Programming repose sur quatre valeurs fondamentales[65] :

La communication :
C'est un principe fondamental entre tous les acteurs du projet pour éviter les problèmes. Les tests, la programmation en binôme et le jeu du planning obligent les développeurs, les décideurs et les clients à communiquer. Si un manque apparait, un coach se charge de l'identifier et de remettre ces personnes en contact.

La simplicité :

La façon la plus simple d'arriver au résultat est la meilleure. Anticiper les extensions futures est une perte de temps. Une application simple sera plus facile à faire évoluer.

feedback :

Le retour d'information est primordial pour le programmeur et le client. Les tests unitaires indiquent si le code fonctionne. Les tests fonctionnels donnent l'avancement du projet. Les livraisons fréquentes permettent de tester les fonctionnalités rapidement.

Le courage :

Certains changements demandent beaucoup de courage. Il faut parfois changer l'architecture d'un projet, jeter du code pour en produire un meilleur ou essayer une nouvelle technique. Le courage permet de sortir d'une situation inadaptée. C'est difficile, mais la simplicité, le feedback et la communication rendent ces tâches accessibles.

III.2 Processus de composition des services Cloud :

La (figureIII.1) illustre la procédure suivie afin de permettre la composition des services Cloud.

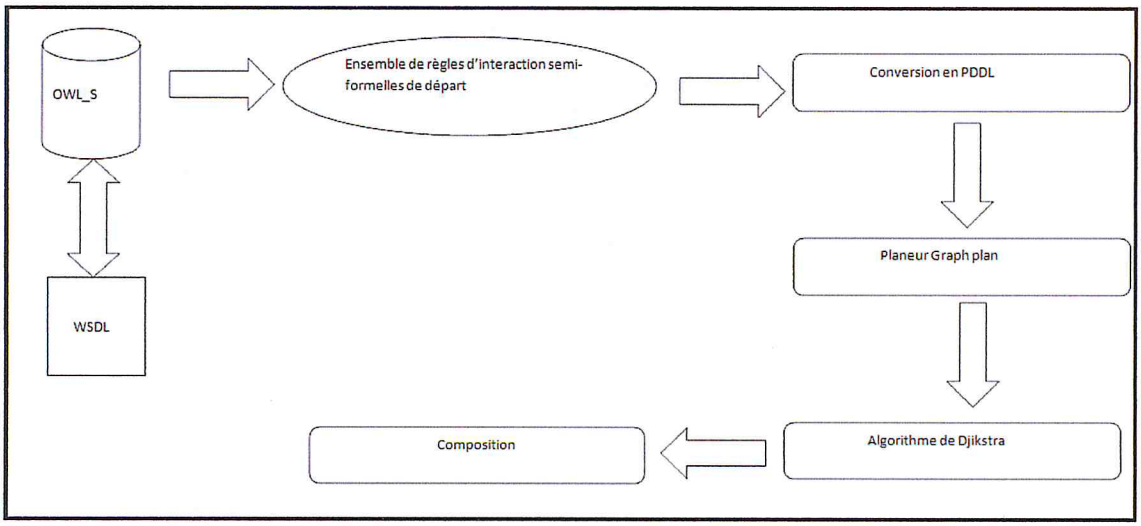


FIGURE III.1 – *Processus de réalisation de composition des services Cloud*

Afin de comprendre le fonctionnement de ce processus nous allons l'expliquer étape par étape dans ce qui suit :

III.2.1 Description des Saas :

Au départ, nous avons commencé à décrire les différents Saas en utilisant l'ontologie OWLS. OWLS est creé par le groupe de DAML[69], et se compose de trois

parties. Première, le profil de service qui décrit les capacités du service. Ensuite, le modèle de service qui décrit comment le service fonctionne intérieurement. Enfin, un sol de service qui décrit comment accéder au service. OWLS ontologie est utile parce qu'il fournit un mécanisme uniforme pour décrire la sémantique d'un service Web.

La (figureIII.2) comporte trois parties principales de l'ontologie OWL-S : le profil de service(profile), le modèle de processus(process model) et Le sol de service(grounding).

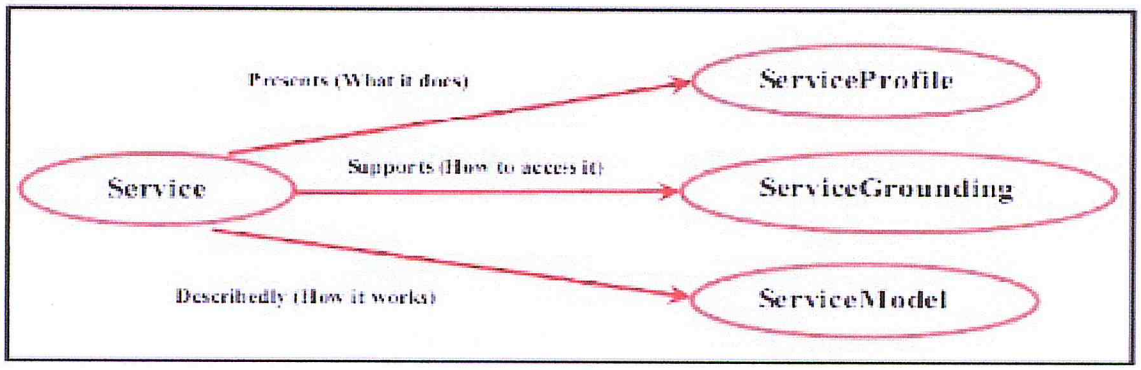


FIGURE III.2 – *S*tructure générale de l'ontologie OWL-S [71]

Saas	Cloud Provider	Prerequisite
Saas1	Cloud1	Saas6
Saas2	Cloud1	Saas5
Saas3	Cloud1	/
Saas4	Cloud2	/
Saas5	Cloud2	/
Saas6	Cloud3	Saas4
Saas7	Cloud3	Saas1
Saas8	Cloud3	Saas3

TABLE III.2 – *Exemple d'interopérabilité entre trois Cloud*

▷ **Processus atomiques :** On commence à créer des instances de la classe Process. On a l'intention de décrire quatre différents d'activités pour notre processus : ExecuterServices(pour l'éxecution des services qui n'ont pas des pré- requis), DemandeService1 , DemandeServices2 et DemandeServices3. Dans ce cas, nous avons décidé les entrées et les sorties pour chacun des processus atomiques.

• **Description des activités :**

1)- ExecuterServices : Nous avons commencé par l'exécution des Saas(Saas3, Saas4, Saas5) qui n'ont pas des prérequis et ne demandent aucune sortie d'un autre service dans son entrée pour qu'il puisse d'êtres exécuter La (figureIII.3).

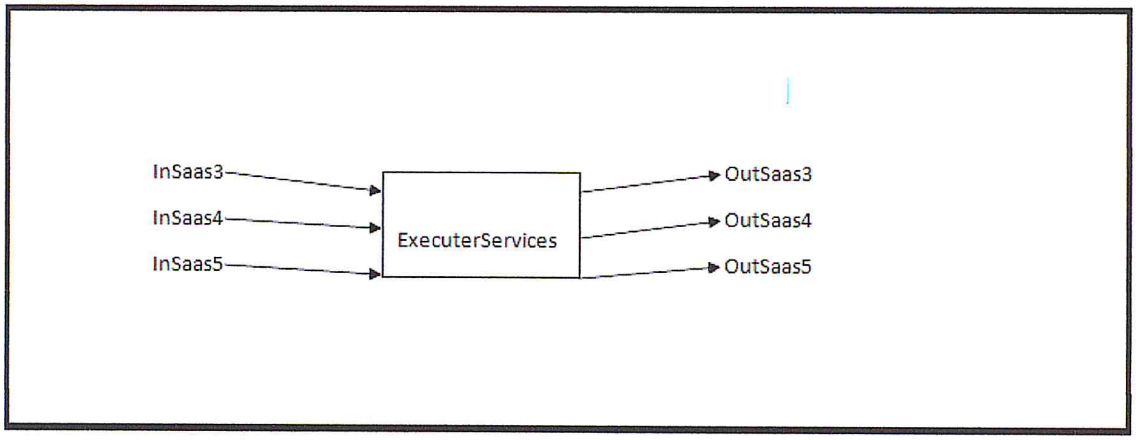


FIGURE III.3 – *L*es entrées et les sorties de processus atomique ExecuterServices

2)- DemandeService1 : Cette activité consiste à exécuter les Saas(Saas8, Saas6, Saas2) qui ils ont comme entrées les sorties résultant d'après l'exécution de l'activité 'ExecuterServices' La (figureIII.4).

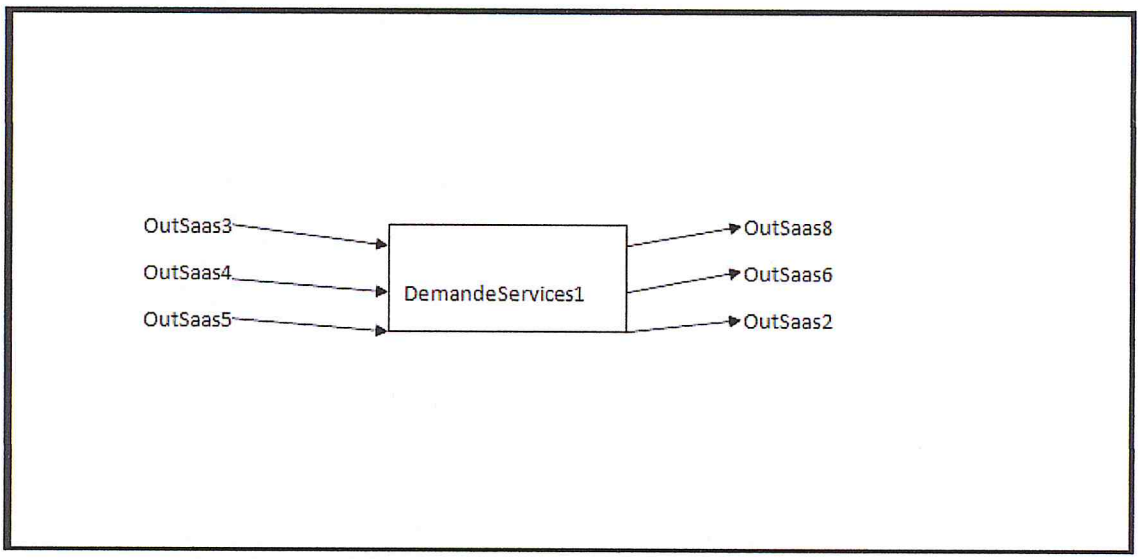


FIGURE III.4 – *L*es entrées et les sorties de processus atomique DemandeService1

3)- DemandeService2 : La (figureIII.5) montre l'exécution de Saas1 aprés la réception de la sortie de Saas6.

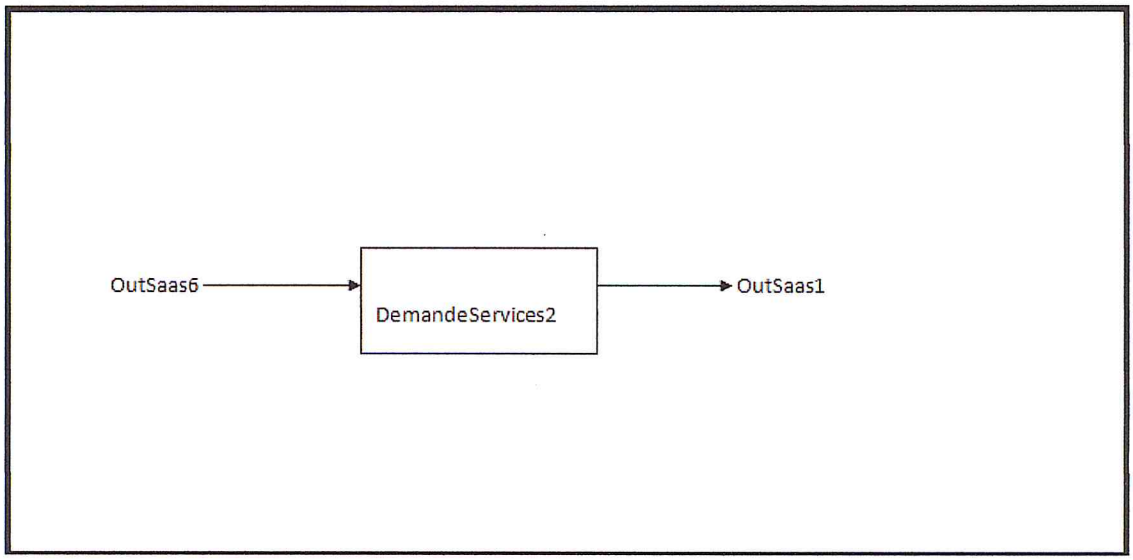


FIGURE III.5 – *L*es entrées et les sorties de processus atomique DemandeService2

4)- DemandeService3 : La (figureIII.6) montre l'exécution de Saas7 qui il a comme entrée la sortie résultant d'après l'exécution de l'activité 'DemandeService2'.

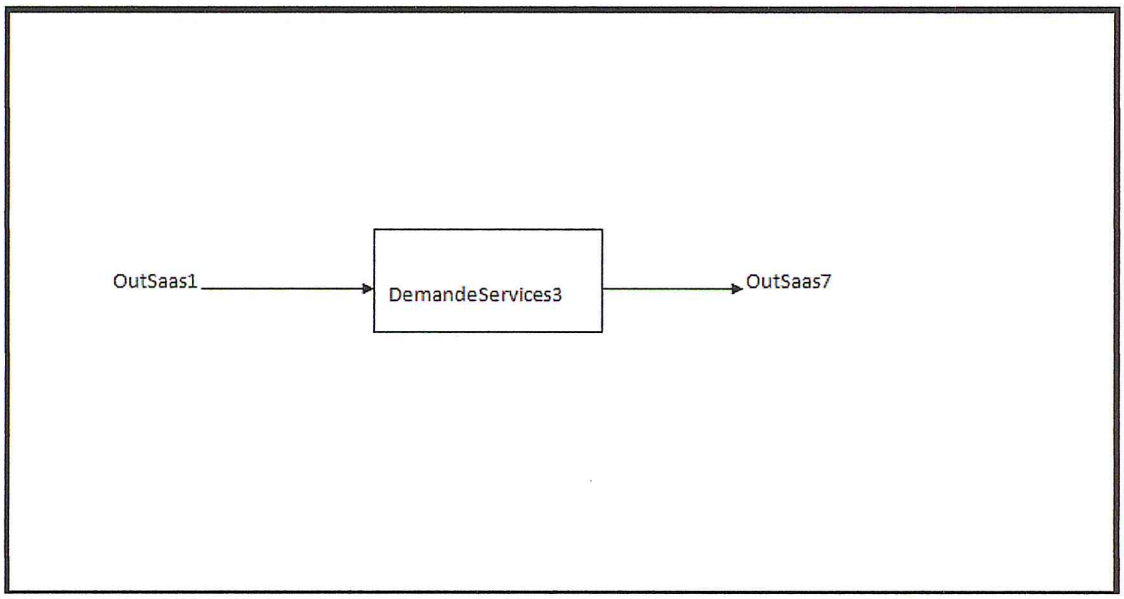


FIGURE III.6 – *L*es entrées et les sorties de processus atomique DemandeService3

En autre, nous avons une présentation qui montre les relations entre les différentes composantes d'un service OWL-S sous forme de graphique.

La vue d'ensemble graphique de notre travail est illustré dans la (figure III.7). Elle montre tous les services, les profils et les processus de haut niveau. Cela donne un bon aperçu de quelle manière les différents éléments sont relié.

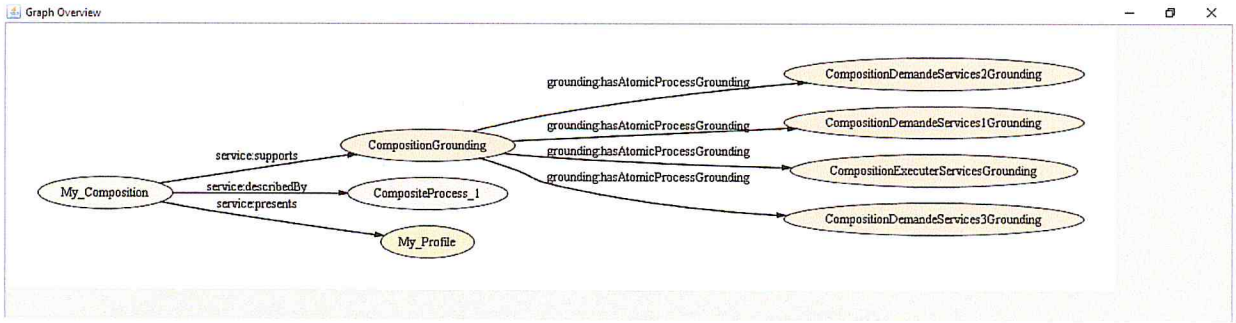


FIGURE III.7 – *Graphe overview*

III.2.2 Ensemble de règles d'interaction semi- formelles de départ :

La construction des règles d'interaction semi-formelles est une étape très importante dans notre processus, elle nous permet de donner un aperçu sur la façon dont laquelle les interactions devraient être établies, afin de proposer une solution adaptable à nos prévisions.

— Etat(0) correspond à l'état du service avant son exécution.
— Etat(1) correspond à l'état du service avant sa mise en attente.

1)- Un service S Etat(0) besoin d'un service S'→ si S' est libre → exécuter S' → envoyer S' après son exécution à S.

2)- Un service S Etat(1) besoin de S' → si S' est occupé → mise en attente du S.

3)- Un service S Etat(0) dans Cloud C→ si C est libre → exécuter S.

4)- Un service S Etat(1) dans Cloud C → si C est occupé → mise en attente du S.

III.2.3 Règles semi-formelles traduite en langage de prédicats :

Les objets :

Cloud : C1 , C2, C3.
services : S1 , S2 , S3 , S4 , S5 , S6 , S7 , S8.

Les prédicats :

On(X,Y)	- Vrai si X est le service placé dans le Cloud Y.
Request(Xn,Xm)	- Vrai si Xn est le service demandé par le service Xm.
Check(X)	- Vrai si l'état du Cloud X est vérifié.
Free(X)	- Vrai si X est un Cloud libre (inactif).
Execute(X,Y)	- Vrai si X est le service à exécuter par le Cloud Y.
Wait(X,Y)	- Vrai si X est le service mit en attente par le Cloud Y.
Send(X,Y)	- Vrais si X est le service a envoyé par le Cloud Y.
Resive(X,Y)	- Vrai si X est le service acquis par le Cloud Y.

Etat initial :

On(S6, C3), On(S5, C2), On(S4, C2), On(S1, C1), On(S7, C3), On(S8, C3), On(S2 C1), on(S3,C1) sont vrais.
Request (S6, S1), Request(S5, S2), Request(S4, S6), Request(S1, S7), Request(S3, S8) sont vrais.

Etat final :

Execute(S1, C1), Execute(S2, C1), Execute(S3, C1), Execute(S4, C2), Execute(S5, C2), Execute(S6, C3), Execute(S7, C3), Execute(S8, C3)sont vrais.

Les actions :

1. **Vérification de l'état des services :**

   Description : les services doivent être vérifiés après la réception des demandes.
   Précondition : On(S6, C3), On(S5, C2), On(S4 ,C2), On(S1, C1), On(S3, C1), Request(S6, S1), Request(S5, S2), Request(S4, S6) ,Request(S1, S7), Request(S3, S8) sont vrais.
   effet : Cheek(S6), Cheek(S5), Cheek(S4), Cheek(S1), Cheek(S3), (Free(S6)ou (¬Free(S6)), (Free(S5)ou ¬Free(S5)), (Free(S4)ou ¬Free(S4)), (Free(S1)ou ¬Free(S1)), (Free(S3)ou ¬Free(S3)) sont vrais.

2. **Exécution des services demandés :**

   Description : les services demandés sont exécutés s'ils sont dans un état libre.
   précondition : On(S4, C2),Request(S4, S6), Free(S4),On(S6, C3), Request(S6, S1), Free(S6), On(S1 ,C1), Request(S1, S7),Free(S1), On(S3, C1),Request(S3, S8),Free(S3), On(S5, C2), Request(S5, S2), Free(S5) sont vrais.
   effet : Execute(S4, C2), Output(S4, C2), (¬Free(S4)), Execute(S6, C3), Output(S6 , C3), (¬Free(S6)), Execute(S1, C1), Output(S1, C1), (¬Free(S1)), Execute(S3, C1), Output(S3, C1), (¬Free(S3)), Execute(S5, C2), Output(S5, C2), (¬Free(S5)). sont vrais.

3. **Mise en attente des services demandés :**

   Description : les services demandés sont mise en attente s'ils ne sont pas libres.

   précondition : On(S4, C2), (¬Free (S4)), Request(S4, S6), On(S6, C3), (¬Free(S6)), Request(S6, S1), On(S1, C1), (¬Free(S1)), Request(S1, S7), On(S3,C1), (¬Free (S3)), Request(S3, S8), On(S5, C2), (¬Free(S5)), Request(S5, S2) sont vrais.
   effet : On(S4, C2), Wait(S4, C2), On(S6, C3), Wait(S6, C3), On(S1, C1), Wait(S1, C1), On(S3, C1), Wait(S3, C1), On(S5, C2), Wait(S5, C2) sont vrais.

4. **Envoi des services exécutés :**
   Description : les services exécutés sont envoyés aux services intéressés.
   précondition : On(S4, C2), Execute(S4, C2), Output(S4, C2), On(S6, C3), Execute(S6, C3), Output(S6, C3), On(S1, C1), Execute(S1, C1), Output(S1, C1), On(S3, C1), Execute(S3, C1), Output(S3, C1), On(S5, C2), Execute(S5, C2), Output(S5, C1) sont vrais.
   effet : Free(S4), Send(S4, C2), Input(S4, S6), (¬Execute(S4, C2)), (¬Output(S4, C2)), Free(S6), Send(S6, C3), Input(S6, S1), (¬Execute(S6, C3)), (¬Output(S6, C3)), Free(S1), Send(S1, C1), Input(S1, S7), (¬Execute(S1, C1)), (¬Output(S1, C1)), Free(S3), Send(S3, C1), Input(S3, S8), (¬Execute(S3, C1)), (¬Output(S3, C1)), Free(S5), Send(S5, C2), Input(S5, S2), (¬Execute(S5, C2)), (¬Output(S5, C2)) sont vrais.

5. **Réception des services envoyés :**
   Description : Les services envoyés sont reçus par les services intéressés.
   précondition : Send(S4, C2), Input(S4, S6), (¬Execute(S4, C2)), Send(S6, C3), Input(S6, S1), (¬Execute(S6, C3)), Send(S1, C1), Input(S1, S7), (¬Execute(S1, C1)), Send(S3, C1), Input(S3, S8), (¬Execute(S3, C1)), Send(S5, C2), Input(S5, S2), (¬Execute(S5, C2)) sont vrais. **effet :** Resive(S4, S6), Execute(S6, C3), Resive(S6, S1), Execute(S1, C1), Resive((S1, S7), Execute(S7, C3), Resive(S3, S8), Execute(S8, C3), Resive(S5, S2), Execute (S2, C1) sont vrais.

Après la création des règles d'interaction semi-formelles qui mène à la résolution de notre problème, nous avons choisir l'algorithme de planification Graphplan qui prend en considération l'ensemble de ses règles.
L'algorithme de planification Graphplan a été déjà utilisé dans la composition de web services en apportant de bon résultat. Notre objectif est d'utiliser cet algorithme pour la technologie du Cloud Computing dans le but de résoudre le problème de la composition des services Cloud. Mais avant cela, nous devons convertir l'ensemble de règles d'interaction semi-formelles en langage PDDL qui fournit une syntaxe permettant au Graphplan de la prendre en charge.

III.2.4 Conversion en PDDL des règles d'interaction semi-formelles :

PDDL (Planning Domain Definition Language), est un langage de codage standard pour les tâches de planification "classiques" [72], il permet une représentation claire d'un problème de planification. En effet, il donne la possibilité de représenter les tâches de planification grâce à des variables, des prédicats et des connecteurs logiques.
Le problème de planification est séparé en deux parties, la description du domaine et la description du problème[73].

1. **Le fichier PDDL de problème :** il contient les objets qui peuvent être utilisés à la place des variables, une description de l'état initial et et les résultats[72].

▷ **Fichier de problème :**

```
;; problem file: cloudcomposition-prob.pddl
(define (problem cloudcomposition-prob)
(:domain cloudcomposition)
(:objects S1 S2 S3 S4 S5 S6 S7 S8 C1 C2 C3)
(:init (On S4 C2) (On S6 C3)(On S1 C1) (On S3 C1)(On S5 C2)(Request S4 S6)(Request S6 S1)
(Request S1 S7)(Request S3 S8)(Request S5 S2))
(:goal(and(Execute S1 C1)(Execute S2 C1)(Execute S3 C1)(Execute S4 C2)
(Execute S5 C2)(Execute S6 C3)(Execute S7 C3)(Execute S8 C3))))

```

FIGURE III.8 – *Fichier de problème*

▷ **Fichier de domaine :**

```
(define (domain cloudcomposition)
(:requirements : strips)
(:predicates
        (On ?x ?y)
       (Requeste ?xn ?xm)
       (Cheek ?x)
       (Free ?x)
        (Wait ? x ?y)
         (Input ?x ?y)
         (Output ?x ?y)
         (Execute ?x ?y)
        (Send ?x ?y)
       (Resive ?xn ?xm)
)
(: action CheekService
             : parameters(?S1 ?S2 ?S3 ?S4 ?S5 ?S6 ?S7 ?S8 ?C1 ?C2 ?C3)
             :precondition(and(On ?S6 ?C3)(On ?S5 ?C2)(On ?S4 ?C2)(On ?S1 ?C1)(On ?S3 ?C1)
             (Request S6 S1 )(Request S5 S2)(Request S4 S6)(Request S1 S7)(Request S3 S8))
             :effect (and (Cheek ?S6) (Cheek ?S5) (Cheek ?S4) (Cheek ?S1) (Cheek ?S3) (Free ?S6)
             (Free ?S5)(Free ?S4)(Free ?S1)(Free ?S3)(not(Free ?S6))(not(Free ?S5))(not(Free ?S4))
             (not(Free ?S1))(not(Free ?S3))))
(: action ExecuteService
           : parameters(?S1 ?S2 ?S3 ?S4 ?S5 ?S6 ?S7 ?S8 ?C1 ?C2 ?C3)
           :precondition(and(On ?S4 ?C2)( Request ?S4 ?S6 )(Free ?S4)(On ?S6 ?C3)( Request ?S6 ?S1 )(Free ?S6)
           (On ?S1 ?C1)(Request ?S1 ?S7 )(On ?S3 ?C1)(Request ?S3 ?S8)(Free ?S3)(Free ?S1)(On ?S5 ?C2)(Request ?S5 ?S2)
           (Free ?S5))
           :effect (and(Execute ?S6 C3)(Output ?S6 C3)(not(Free ?S6))(Execute ?S5 C2)(Output ?S5 C2)(not(Free ?S5))
           (Execute ?S4 C2)(Output ?S4 C2)(not(Free ?S4))(Execute ?S1 C1)(Output ?S1 C1)(not(Free ?S1))(Execute ?S3 C1)
           (Output ?S1 C1)(not(Free ?S1)))
```

```
(: action SendService
            : parameters(?S1 ?S2 ?S3 ?S4 ?S5 ?S6 ?S7 ?S8 ?C1 ?C2 ?C3)
            :precondition(and(On ?S4 ?C2)(Execute ?S4 C2)(Output ?S4 C2)(On ?S6 ?C3)(Execute ?S6 C3)(Output ?S6 C3)
            (On ?S1 ?C1)(Execute ?S1 C1)(Output ?S1 C1) (On ?S3 ?C1) (Execute ?S3 C1)(Output ?S3 C1))
            (On ?S5 ?C2) (Execute ?S5 C2)(Output ?S5 C2))
            :effect (and(Free ?S4)(Send ?S4 C2)(Input ?S4 ?S6)(not(Execute ?S4 ?C2))(not(Output ?S4 ?C2))
            (Free ?S6)(Send ?S6 C3)(Input ?S6 ?S1)(not(Execute ?S6 ?C3))(not(Output ?S6 ?C3))
            (Free ?S1)(Send ?S1 C1)(Input ?S1 ?S7)(not(Execute ?S1 ?C1))(not(Output ?S1 ?C1))
            (Free ?S3)(Send ?S3 C1)(Input ?S3 ?S8)(not(Execute ?S3 ?C1))(not(Output ?S3 ?C1))
            (Free ?S5) (Send ?S5 C2)(Input ?S5 ?S2)(not(Execute ?S5 ?C2))(not(Output ?S5 ?C2))))
(: action WaitService
            :parameters(?S1 ?S2 ?S3 ?S4 ?S5 ?S6 ?S7 ?S8 ?C1 ?C2 ?C3)
            :precondition(and(On ?S4 ?C2)(not(Free ?S4)( Request ?S4 ?S6)(On ?S6 ?C3)(not(Free ?S6)( Request ?S6 ?S1)
            (On ?S1 ?C1)(not(Free ?S1)(Request ?S1 ?S7)(On ?S3 ?C1)(not(Free ?S3)(Request ?S3 ?S8)(On ?S5 ?C2)
            (not(Free ?S5)(Request ?S5 ?S2))
            :effect(and(On ?S4 ?C2)(Wait ?S4 ?C2)(On ?S6 ?C3)(Wait ?S6 ?C3)(On ?S1 ?C1)(Wait ?S1 ?C1)(On ?S3 ?C1)(Wait ?S3 ?C1)
            (On ?S5 ?C2) (Wait ?S5 ?C2)))
(: action ResiveService
            : parameters(?S1 ?S2 ?S3 ?S4 ?S5 ?S6 ?S7 ?S8 ?C1 ?C2 ?C3)
            :precondition(and (Send ?S4 C2)(Input ?S4 ?S6)(not(Execute ?S4 ?C2))(Send ?S6 C3)(Input ?S6 ?S1)(not(Execute ?S6 ?C3))
            (Send ?S1 C1)(Input ?S1 ?S7)(not(Execute ?S1 ?C1))(Send ?S3 C1)(Input ?S3 ?S8)(not(Execute ?S3 ?C1))
            (Send ?S5 C2)(Input ?S5 ?S2)(not(Execute ?S5 ?C2)))
            :effect(and(Resive ?S4 ?S6)(Execute ?S6 ?C3)(Resive ?S6 ?S1)(Execute ?S1 ?C1)(Resive ?S1 ?S7)(Execute ?S7 ?C3)
            (Resive ?S3 ?S8)(Execute ?S8 ?C3)(Resive ?S5 ?S2)(Execute ?S2 ?C1)
```

FIGURE III.9 – *Fichier de domaine*

III.2.5 Planificateur Graphplan :

III.2.5.1 GraphPlan classique :

Au départ, nous avons utilisé le GraphPlan classique en prenant en compte les relations d'exclusions mutuelles entre les différents Saas. Supposons que nous avons S1, S2, S3, S4, S5, S6, S7, S8 les services correspondant aux trois Clouds, en prenant en compte le problème de composition entre ces services via la planification.

Le but de cette planification est de permettre l'interaction entre ces services de sorte que les huits service sont exécutés aprés la réception des demande(tableauIII.2 présente pour chaque service son emplacement et son prerequisite).

▷ ***Le graphe de synthèse :***

La (figureIII.10) représente le graphe de synthèse, les nœuds noir sont les 'no-op' et les trais continus sont les effets ajoutés.

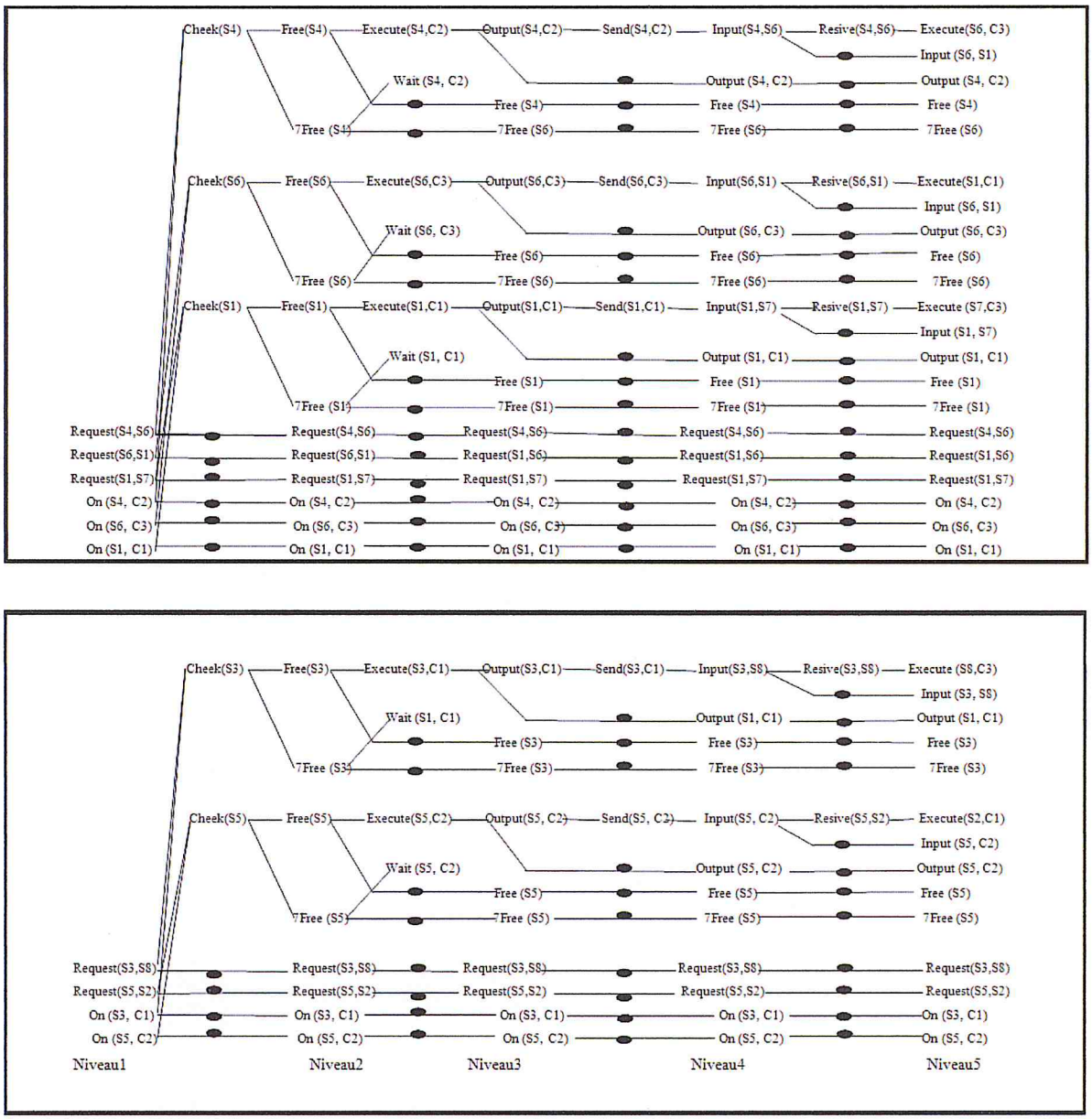


FIGURE III.10 – *Graphe de synthèse*

▷ **Les relations d'exclusions mutuelles entre noeuds :**

En formulant le graphe de synthèse, nous allons détecté les relations d'exclusions mutuelles entre les noeuds d'actions et de propositions.

▷ **Les actions exclusives :**

Deux actions sont 'mutuellement exclusives' dans un même niveau d'actions, si aucun plan valide ne peut les contenir simultanément dans ce même niveau. Deux actions 'a' et 'b' sont marquées exclusives dans un niveau d'actions (i), si on a une des deux conditions suivantes[74] :

• La rivalité : il existe au moins une précondition de 'a' et une précondition de

'b' qui sont mutuellement exclusives dans un niveau de propositions donné.
• L'interférence (non indépendance) : chacune d'entre elles supprime la précondition ou la post-condition de l'autre.

Dans notre cas on a :

1. Execute(S6,C3) et Wait(S6,C3) sont mutuellement exclusive, car la première a besoin de Free(S6) dans sa précondition et la deuxième nécessite ¬Free(S6) dans sa précondition, les deux proposition Free(S6) et ¬Free(S6) sont mutuellement exclusives a un même niveau.
2. Execute(S5,C2) et Wait(S5,C2) sont mutuellement exclusive, car la première a besoin de Free(S5) dans sa précondition et la deuxième nécessite ¬Free(S5) dans sa précondition, les deux proposition Free(S5) et ¬Free(S5) sont mutuellement exclusives a un même niveau.
3. Execute(S4,C2) et Wait(S4,C2) sont mutuellement exclusive, car la première a besoin de Free(S4) dans sa précondition et la deuxième nécessite ¬Free(S4) dans sa précondition, les deux proposition Free(S4) et ¬Free(S4) sont mutuellement exclusives a un même niveau.
4. Execute(S1,C1) et Wait(S1,C1) sont mutuellement exclusive, car la première a besoin de Free(S1) dans sa précondition et la deuxième nécessite ¬Free(S1) dans sa précondition, les deux proposition Free(S1) et ¬Free(S1) sont mutuellement exclusives a un même niveau.
5. Execute(S3,C1) et Wait(S3,C1) sont mutuellement exclusive, car la première a besoin de Free(S3) dans sa précondition et la deuxième nécessite ¬Free(S3) dans sa précondition, les deux proposition Free(S3) et ¬Free(S3) sont mutuellement exclusives a un même niveau.
6. On a aussi Send(S6,C3) et Resive(S6,S1) sont mutuellement exclusives dans le niveau 4 car la première a besoin de Output(S6,C3) dans sa précondition et la deuxième nécessite Input(S6,S1) dans sa précondition, les deux propositions Output(S6,C3) et Input(S6,S1) sont mutuellement exclusives a un même niveau.
7. On a aussi Send(S5,C2) et Resive(S5,S2) sont mutuellement exclusives dans le niveau 4 car la première a besoin de Output(S5,C2) dans sa précondition et la deuxième nécessite Input(S5,S2) dans sa précondition, les deux propositions Output(S5,C2) et Input(S5,S2) sont mutuellement exclusives a un même niveau.
8. On a aussi Send(S4,C2) et Resive(S4,S6) sont mutuellement exclusives dans le niveau 4 car la première a besoin de Output(S4,C2) dans sa précondition et la deuxième nécessite Input(S4,S6) dans sa précondition, les deux propositions Output(S4,C2) et Input(S4,S6) sont mutuellement exclusives a un même niveau.
9. On a aussi Send(S1,C1) et Resive(S1,S7) sont mutuellement exclusives dans le niveau 4 car la première a besoin de Output(S1,C1) dans sa précondition et la deuxième nécessite Input(S1,S7) dans sa précondition,

les deux propositions Output(S1,C1) et Input(S1,S7) sont mutuellement exclusives a un même niveau.

10. On a aussi Send(S3,C1) et Resive(S3,S8) sont mutuellement exclusives dans le niveau 4 car la première a besoin de Output(S3,C1) dans sa précondition et la deuxième nécessite Input(S3,S8) dans sa précondition, les deux propositions Output(S3,C1) et Input(S3,S8) sont mutuellement exclusives a un même niveau.

▷ **Les propositions exclusives :**

Deux propositions sont 'mutuellement exclusives' dans un même niveau, si aucun plan valide ne peut les rendre vraie toutes les deux dans ce même niveau[74]. Dans notre graphe de synthèse on a les propositions mutuellement exclusives suivantes :

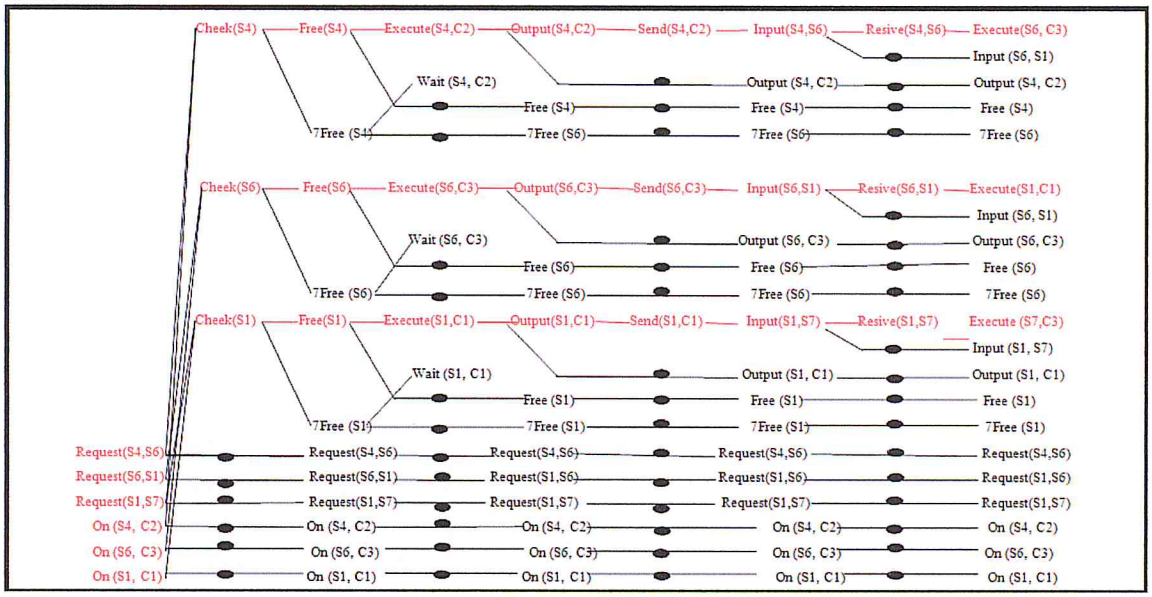
- Free(S6) et ¬Free(S6) sont mutuellement exclusives.
- Free(S5) et ¬Free(S5) sont mutuellement exclusives.
- Free(S4) et ¬Free(S4) sont mutuellement exclusives.
- Free(S1) et ¬Free(S1) sont mutuellement exclusives.
- Free(S3) et ¬Free(S3) sont mutuellement exclusives.
- Output(S6,C3) et Input(S6,S1) sont mutuellement exclusives dans le niveau 4.
- Output(S5,C2) et Input(S5,S2) sont mutuellement exclusives dans le niveau 4.
- Output(S4,C2) et Input(S4,S6) sont mutuellement exclusives dans le niveau 4.
- Output(S1,C1) et Input(S1,S7) sont mutuellement exclusives dans le niveau 4.
- Output(S3,C1) et Input(S3,S8) sont mutuellement exclusives dans le niveau 4.

Graphplan utilise toutes ces régles relatives aux actions et propositions exclusives pour detécter et enregistrer les relations d'exclusion mutuelles entre les noeuds[74].

▷ ***La recherche du plan valide :***

Après la formulation du graphe de synthèse et après l'obtention des buts totalement indépendants, Graphplan commence à rechercher un plan en utili-

sant une stratégie à chainage arrière[74]. Il commence par les objectifs, en prenant en compte les relations d'exclusions mutuelles établies au cours de la phase de création du graphe. A partir du chemin obtenu qui est (marqué en rouge) nous sommes parvenus à atteindre les conditions initiales comme il montre la (figureIII.11) ci dessous.



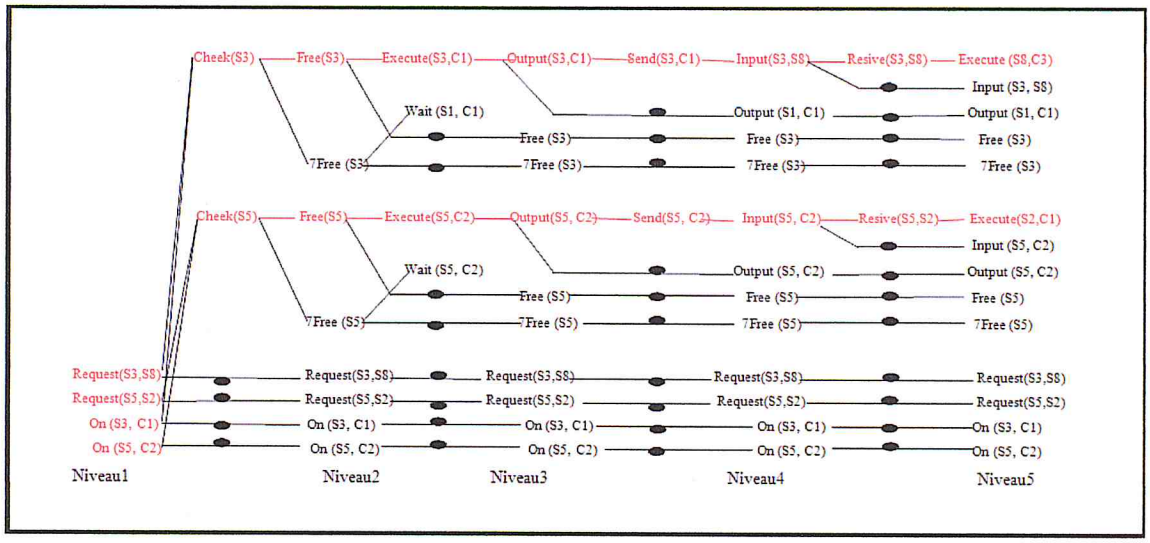


FIGURE III.11 – *R*echerche du plan valide

III.2.5.2 GraphPlan amélioré :

dans cette partie, nous avons utilisé le GraphPlan amélioré pour montrer tous les solutions possibles pour atteindre l'ordonnancement des différents Saas. Supposons que Saa1, Saa2, Saa3, Saa4, Saa5, Saa6 les services destinés au Clouds C1, C2, C3. Le (tableauIII.7) montre l'emplacement de chaque service avec son prerequisite.

Saas	Cloud Provider	Prerequisite
Saa1	Cloud1	
Saa2	Cloud1	Saa1
Saa3	Cloud1	Saa1
Saa4	Cloud2	Saa2
Saa5	Cloud2	Saa2, Saa3
Saa6	Cloud3	Saa5, Saa4

TABLE III.3 – *Etude de cas 2*

▷ ***Construction GraphPlan :***
La (figureIII.12) représente la construction de GraphPlan du tableauIII.3

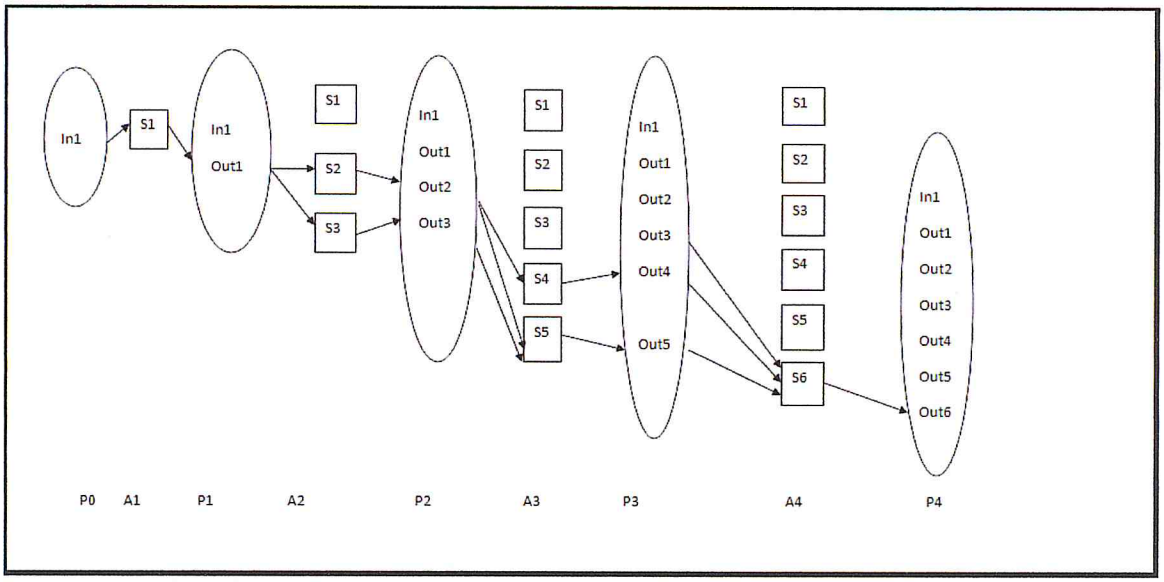


FIGURE III.12 – *C*onstruction GraphPlan

Au début, nous avons commencé par le service (Saa1) qui n'a pas besoin dans son entrés la sortie d'un notre service (niveau P0, A0). ensuite nous avons placé le reste des services selon ses entrés jusqu'à arriver au service(Saa6) au niveau(P4) qui demande dans son entré les sorties de deux services (Saa5 et Saa4).

▷ ***La recherche du plan valide :***
Après la formulation du graphe (figureIII.12), nous avons obtenu les chemins possibles en commençant par Saas1 jusqu'à arrivé au Saas6 qui sont marqués (en vert, en rouge, et en blue) comme il montre la (figureIII.13) ci dessous.

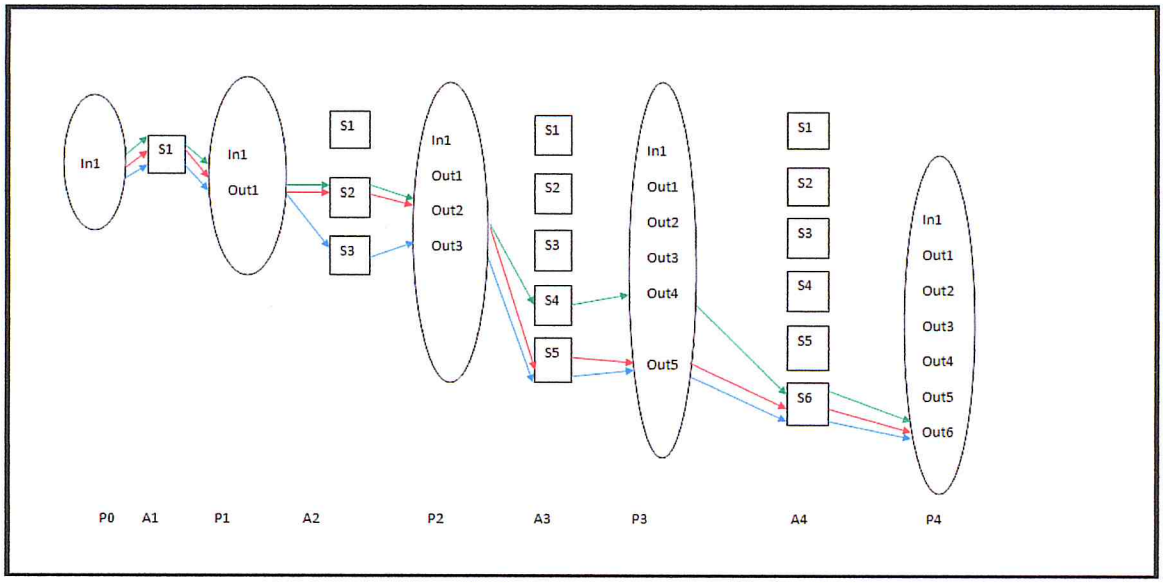


FIGURE III.13 – *R*echerche du plan valide

▷ ***Étape d'optimisation du plan valide :***

Après la recherche du plan valide et l'obtention de plusieurs chemins, nous avons passé à la phase d'optimisation pour minimiser le nombre de composants en utilisant l'algorithme de Djikstra suit d'un critère de distance entre le courtier et les fournisseurs. La (figureIII.14) montre le plus court chemin (marqué en rouge) et l'algorithme général est décrit dans l'annexe C.

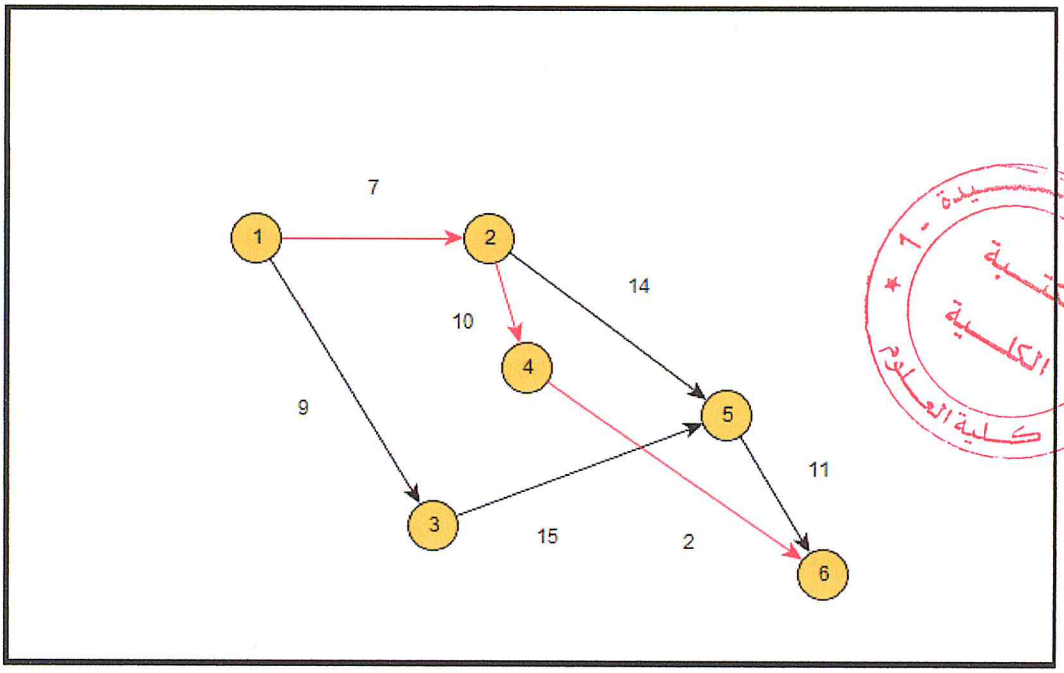


FIGURE III.14 – *S*olution optimale

III.2.6 Composition des services

Cette phase consiste à faire une composition entre différents SaaS en utilisant le langage Bpel(Business Process Execution Language). Bpel est la norme pour l'assemblage d'un ensemble de services discrets dans un flux de processus de bout en bout, réduisant radicalement le coût et la complexité des initiatives d'intégration de processus[76]. La partie expérimentation dans le chapitre suivant montre la réalisation de composition des services.

III.3 Conclusion :

Dans ce chapitre, nous avons présenté un processus pour la modélisation de la composition de services Cloud de type SaaS. L'idée de base est d'utiliser le planificateur Graphplan pour réaliser la planification des interactions des services Cloud en utilisant le langage PDDL, qui est nécessaire pour l'utilisation de Graphplan. Nous avons utilisé aussi l'algorithme de Djikstra pour l'optimisation le nombre de composants. Dans le chapitre suivant, nous allons présenter l'application dans laquelle nous avons implémenté notre solution.

IV

Expérimentation :

IV.1 Introduction :

Dans le chapitre précédent de ce mémoire, nous avons proposé une solution qui assure la composition des services Cloud(SaaS) basée sur un processus bien défini.

Dans ce chapitre nous allons présenter, dans un premier temps, l'environnement de travail où on va choisir le langage de programmation à utiliser ainsi que l'environnement matériel et logiciel dont nous aurons besoin pour passer par la suite aux étapes de la simulation.

IV.2 Langages utilisés :

Pour la réalisation de notre application, nous avons utilisé le langage Java et l'environnement Eclipse.

IV.2.1 Java :

Java est un langage de programmation orienté objet et un environnement d'exécution portables sur plusieurs systèmes d'exploitation tels que : Unix, Windows, Mac OS ou Linux, avec peu ou pas de modifications. Il est créé par James Gosling et Patrick Naughton employés de Sun Microsystems avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld. C'est la plate-forme qui garantit la portabilité des applications développées en Java[77].

Java est basé sur une architecture logicielle très particulière nécessitant une machine virtuelle Java(JVM). Il utilise les notions usuelles de la programmation orientée objet tels que : la notion de classe, d'héritage, de généricité .... Il nécessite un ensemble énorme de bibliothèques standard couvrant de très nombreux domaines. C'est un langage qui présente d'excellentes propriétés de portabilité du code[78].

Ce langage est populaire parce qu'il est la base de la plupart des applications en réseau. Cette technologie est exploitée pour développer et fournir des applications mobiles , des jeux, du contenu Web et des logiciels d'entreprise[77].

IV.2.2 Eclipse :

Eclipse est une communauté open-source(figureIV.1) dont les projets visent à fournir une plateforme de développement, comprenant des espaces de travail, des outils et environnements d'exécution, pour construire, déployer et gérer les applications.

Le Projet Eclipse a été initié par IBM en novembre 2001 et soutenu par un consortium d'acteurs du monde logiciel. La Fondation Eclipse est née en janvier

2004[79].

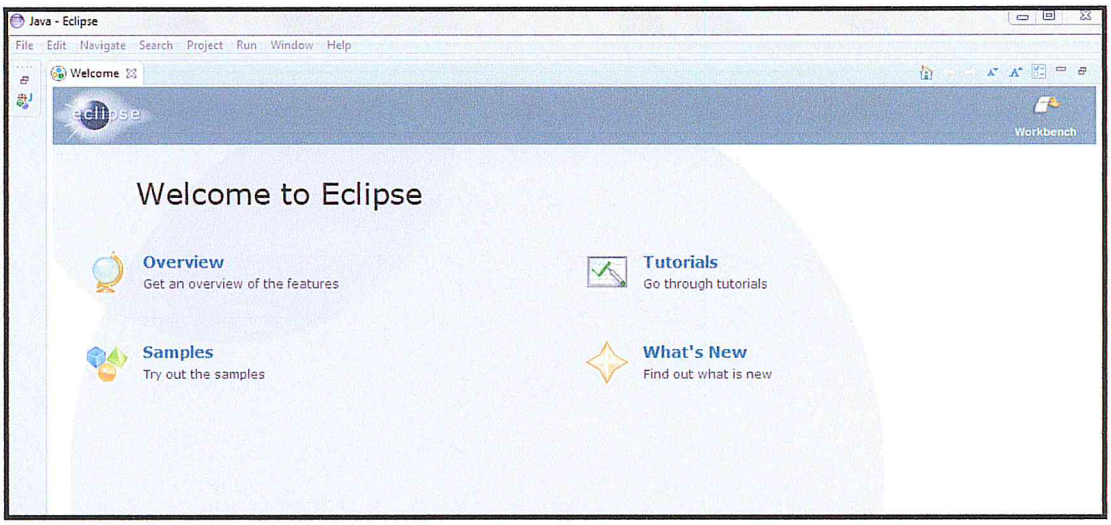


FIGURE IV.1 – *Environnement de développement Eclipse*

IV.3 Description de l'application :

IV.3.1 Interface principale :

Dès le lancement de notre application, la (figureIV.2) est constituée par les composants suivants : Simulation, Liste-Services, GraphPlane, Dijkstra et Composition. Chaque composant contient un ensemble d'items comme il montre ci dessous.

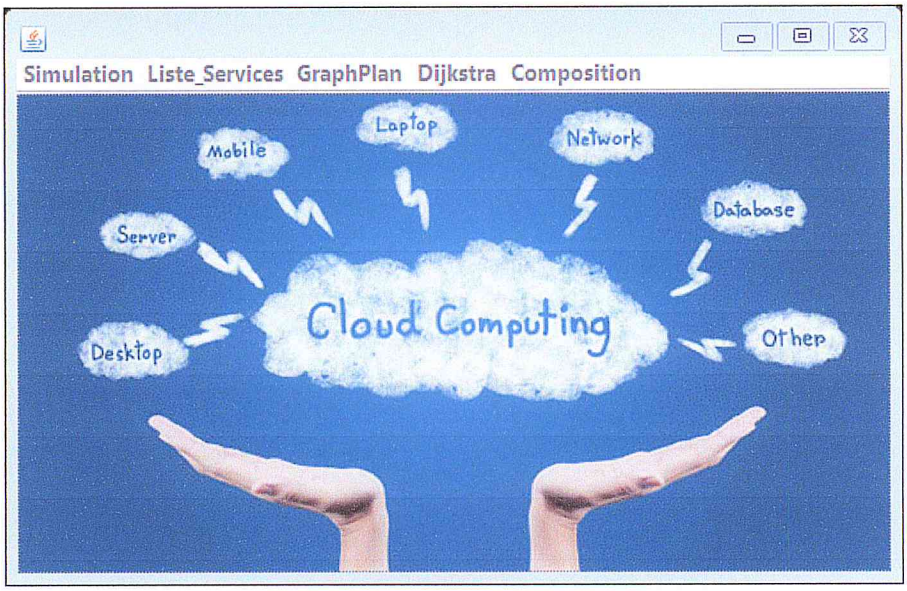


FIGURE IV.2 – *I*nterface principale

IV.3.2 Configuration des paramètres du Cloud :

Nous allons dans cette partie présenter les interfaces de configuration des paramètres de Cloud(La figureIV.3).

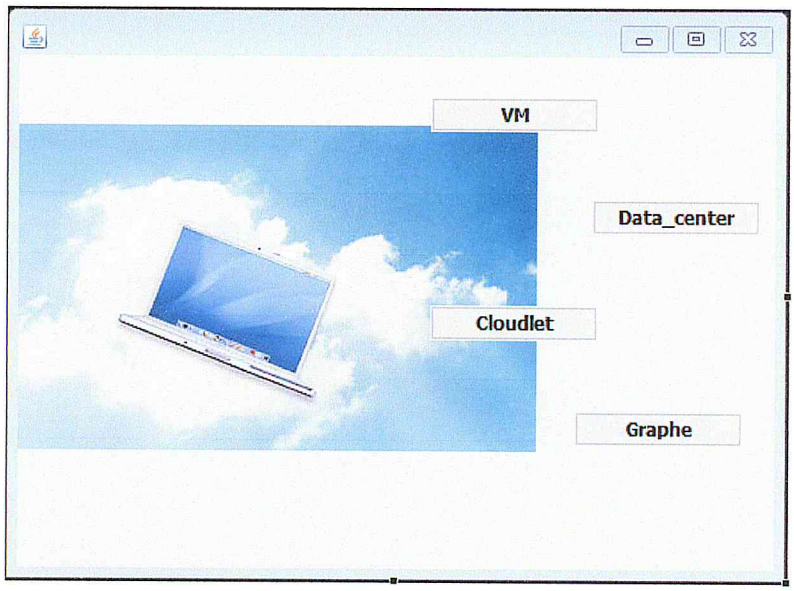


FIGURE IV.3 – *C*onfiguration des paramètres de Cloud

Configuration de la Machine Virtuelle :

Cette étape consiste à configurer les paramètres nécessaires propres à des machines virtuelles correspondantes à un Cloud (figureIV.4), tels que l'identifiant de VM, Mips, la taille de l'image, la RAM et la bande passante.

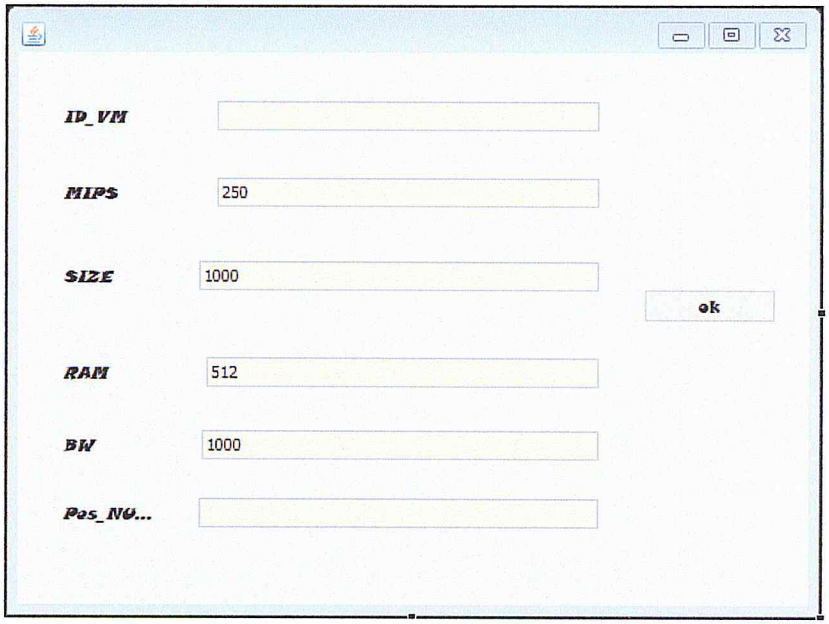


FIGURE IV.4 – *C*onfiguration de la machine virtuelle

La configuration des machines virtuelles représente l'affectation de chaque paramètre des valeurs qui correspondent :

• ID-VM : représente l'identificateur de la VM, il prend une valeur entière positive (ID-VM = 0 ou plus).

• MIPS : représente le nombre de millions d'instructions exécutées par seconde, c'est à dire combien de million d'instructions par seconde, MIPS prend la valeur de 500 et plus selon l'exigence d'application.

• SIZE : représente la taille de l'image qui prend la valeur de 10000 et plus (image size (MB)).

• RAM : prend la valeur de 512 et plus (Vm memory (MB)).

• Bw : La bande passante demandée par VM, elle prend le nombre 1000 et plus (BW = 1000 (MB) et plus).

▷ **Configuration de DataCenter :**

Cette étape consiste à la saisi des paramètres nécessaires pour la configuration de Datacenter (figureIV.5) comme : le nombre d'instructions exécutés par seconde, la taille du stockage de l'host, ainsi que la bande passante.

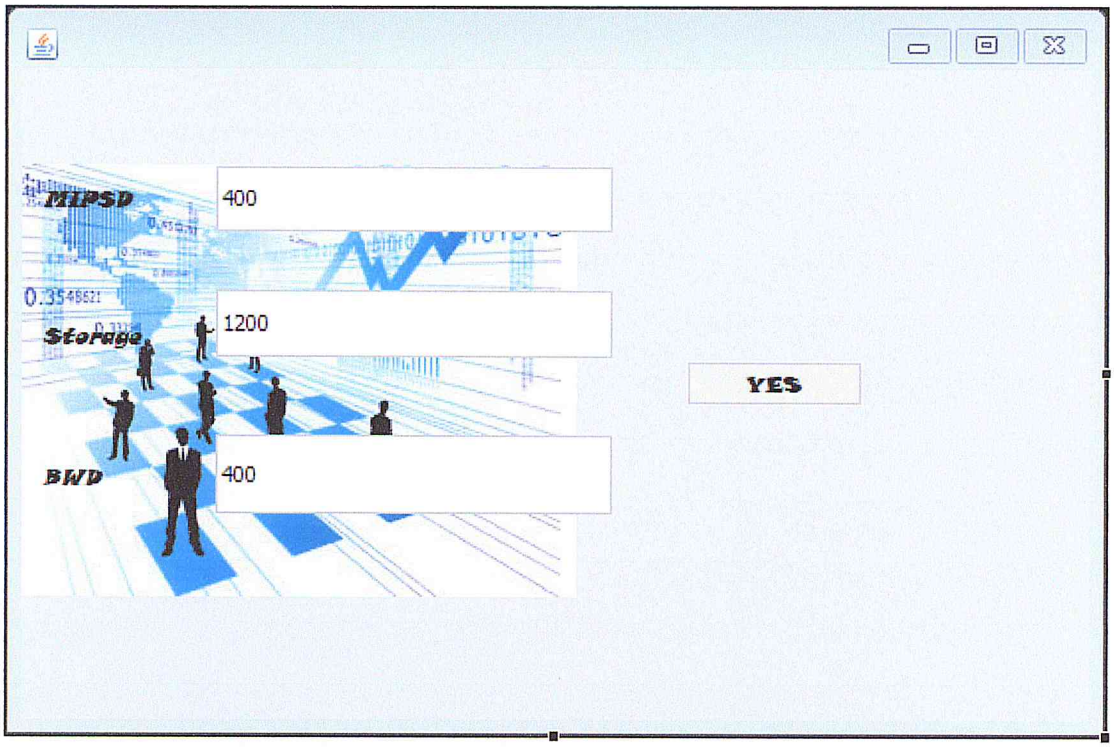


FIGURE IV.5 – *C*onfiguration de Datacenter

La configuration des Datacenter représente l'affectation de chaque paramètre des valeurs qui correspondent :

• MIPSD : représente le nombre de millions d'instructions exécutées par seconde, MIPS prend la valeur de (MIPS= 500 et plus) selon l'exigence de l'application.

• Storage : représente la taille du stockage de l'host qui prend la valeur de 1000000 et plus (host storage (MB)).

• BWD : correspond à la bande passante, elle prend la valeur de 10000 et plus (BW = 10000 (MB) et plus).

▷ **Configuration de Cloudlet :**

Dans cette partie, nous allons configurer le comportement du Cloudlet tels que : l'identificateur du Cloudlet, le nombre de CPU (PEs) utilisé par Cloudlet qui aura un impact sur la quantité MIPs utilisé pour traiter la tâche, fichier d'entrée et de sortie qui a un impact sur les simulations avec le stockage et la longueur du Cloudlet qui est paramétrée par MIPS.

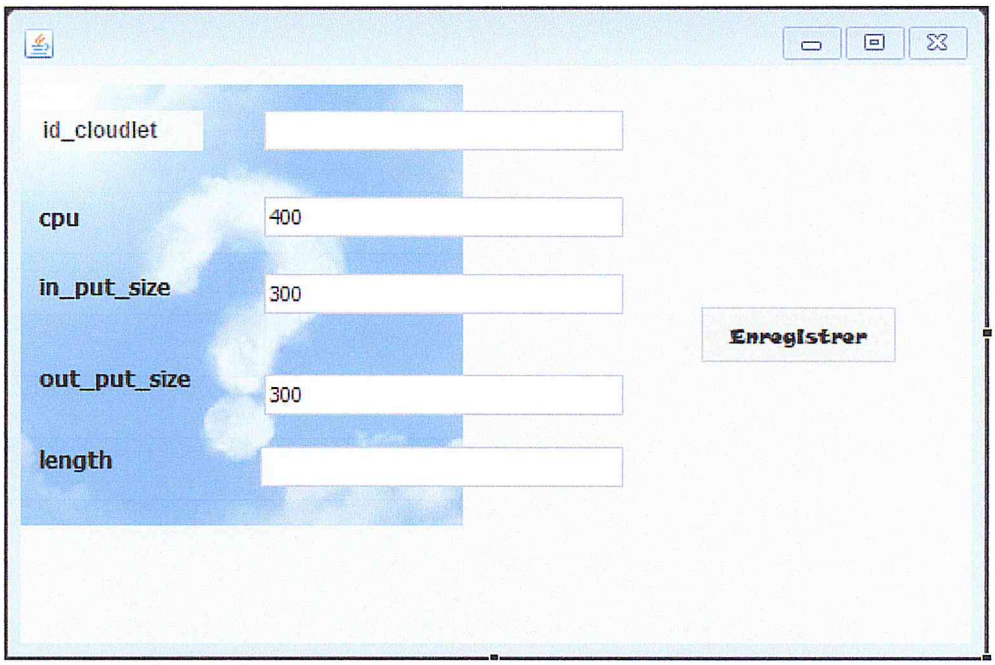


FIGURE IV.6 – *C*onfiguration de Cloudlet

La (figureIV.6) montre les paramètres configurés sur l'onglet Cloudlet :

• id-cloudlet : représente l'identificateur du Cloudlet, il prend une valeur entière positive.

• cpu : le nombre de CPU (PEs) utilisé par Cloudlet, il prend une valeur entière positive.

• in-put-size : la taille du fichier d'entrée du Cloudlet avant l'exécution (la taille du programme + les données en entrée), elle prend un nombre entier positive (exp long fileSize = 300 MB).

• out-put-size : la taille du fichier de sortie du Cloudlet après l'exécution (exp long fileSize = 300 MB).

• length : la taille du Cloudlet à exécuter dans le CloudRessource (exp length = 100000 MB).

IV.3.3 Simulation d'un seul Cloud :

Nous allons dans cette expérimentation simuler un Cloud unique en construisant un graphe qui affiche le développement du temps d'exécution en fonction de la longueur(length(MB)). L'objectif de cette expérience est de doter l'utilisateur des interfaces afin de faciliter la simulation des Cloud en générale.

▷ **Description :**

Dans cette expérience nous allons simuler le Cloudlet en attribuant des valeurs différentes à sa taille (length) qui est initialisée au début par la valeur de 1 MB. En prenant en compte la configuration des autres paramètres du Cloudlet, nous obtiendrons le graphe illustrée dans la (figureIV.7) suivante :

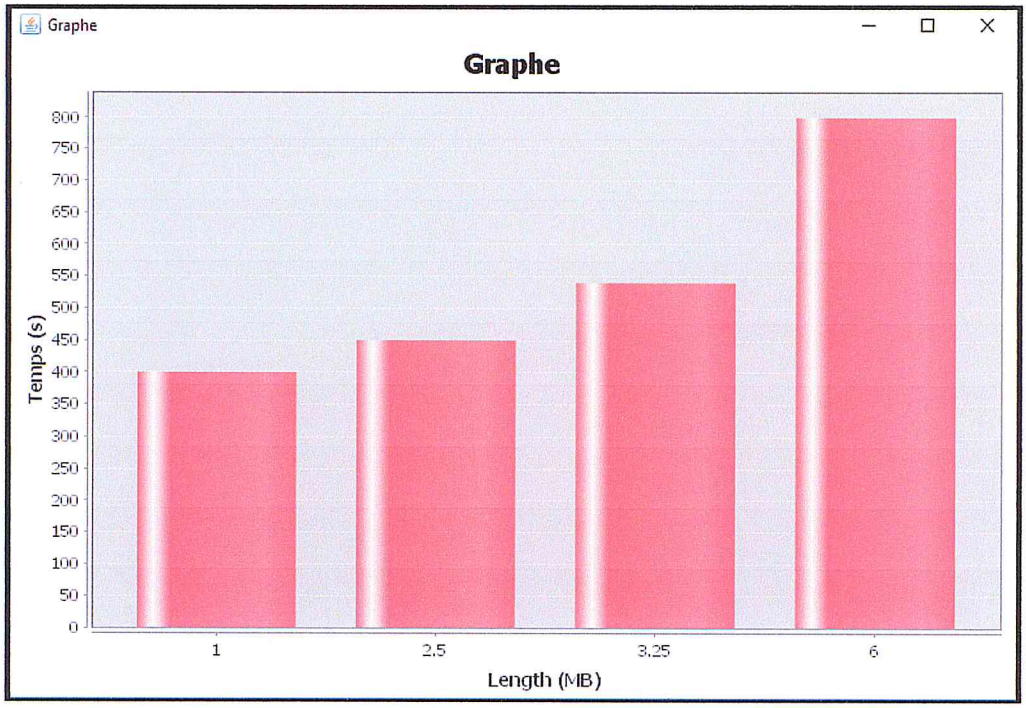


FIGURE IV.7 – *G*raphe de simulation d'un Cloud

IV.3.4 Réalisation de la partie Composition :

Dans cette expérimentation, nous avons ouvrir la porte devant l'utilisateur de sélectionner l'ensemble des services intéressés(la figureIV.8).

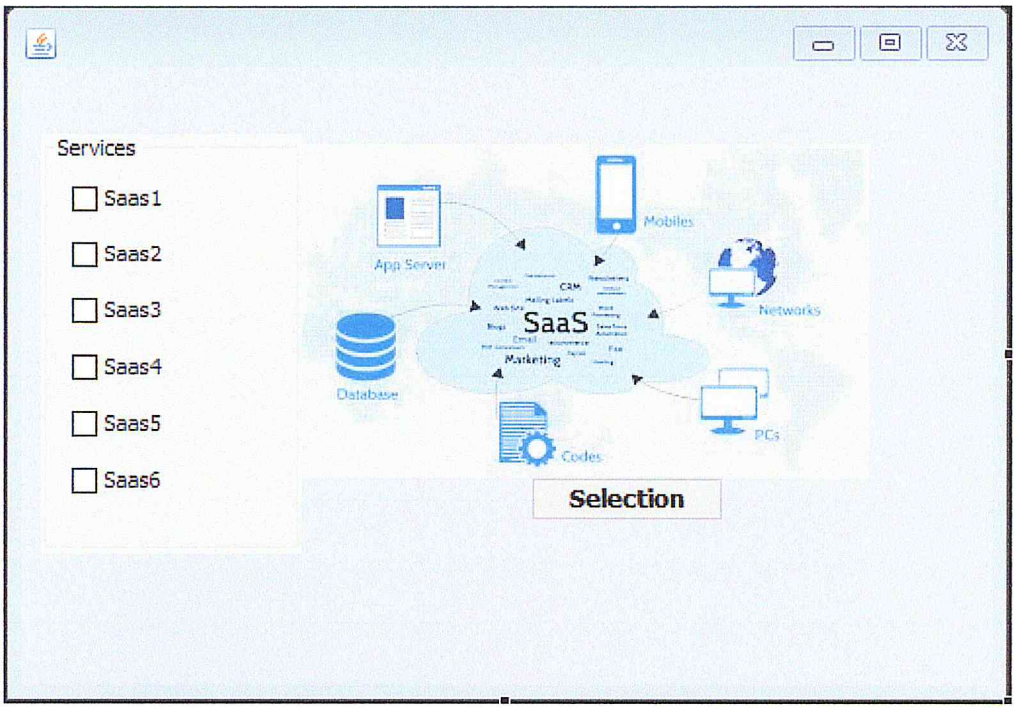


FIGURE IV.8 – *I*nterface utilisateur

Après la sélection des services appropriés, ces derniers sont remplis dans un tableau(figureIV.9) pour passer à la phase de recherche des chemins possibles.

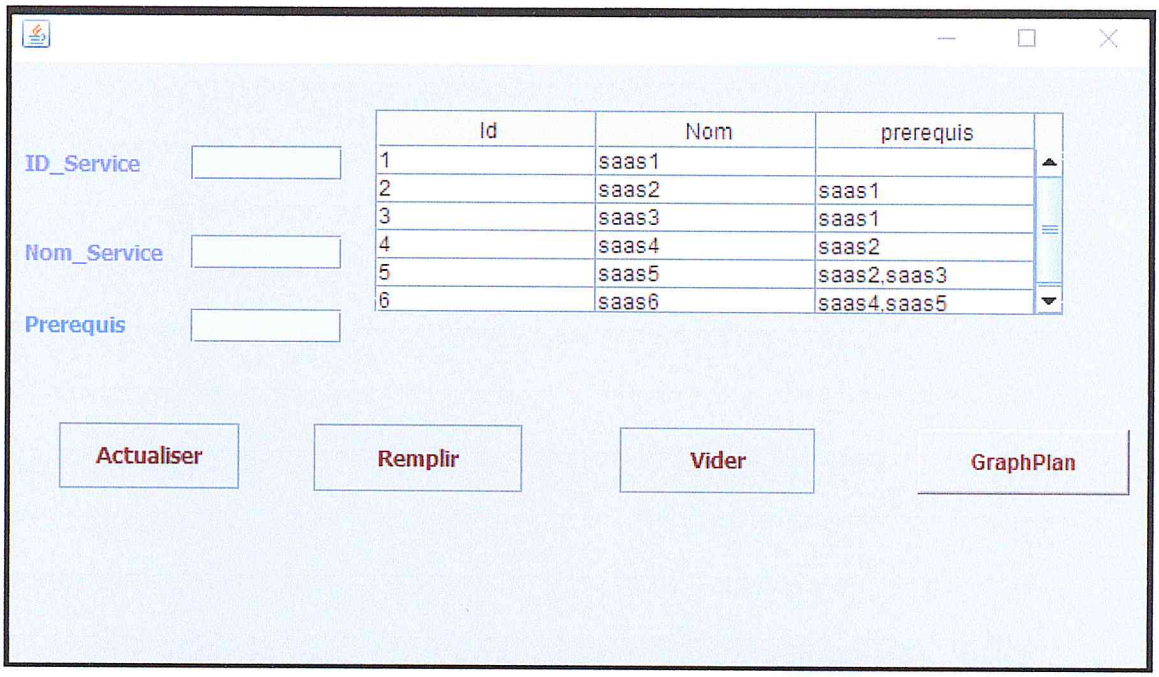


FIGURE IV.9 – *E*nsemble des services sélectionner

La (figureIV.10) et la (figureIV.11) présente les résultats obtenus d'après l'application de GraphPlan.

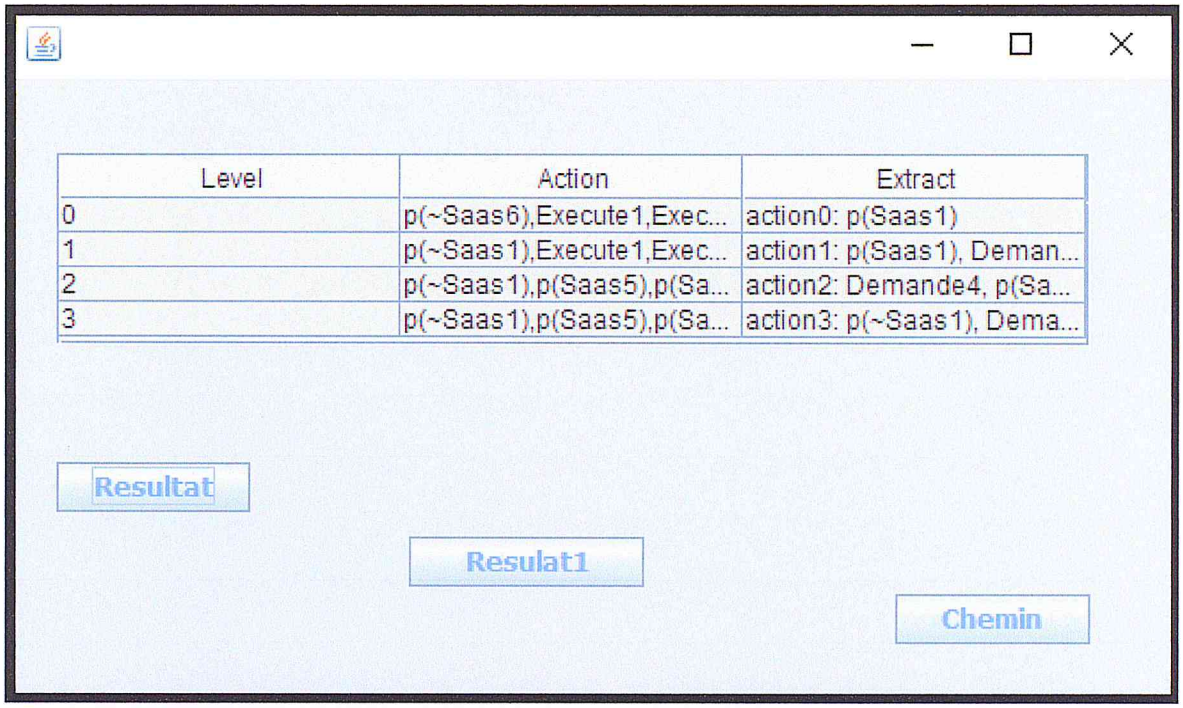


FIGURE IV.10 – *E*nsemble des niveaux

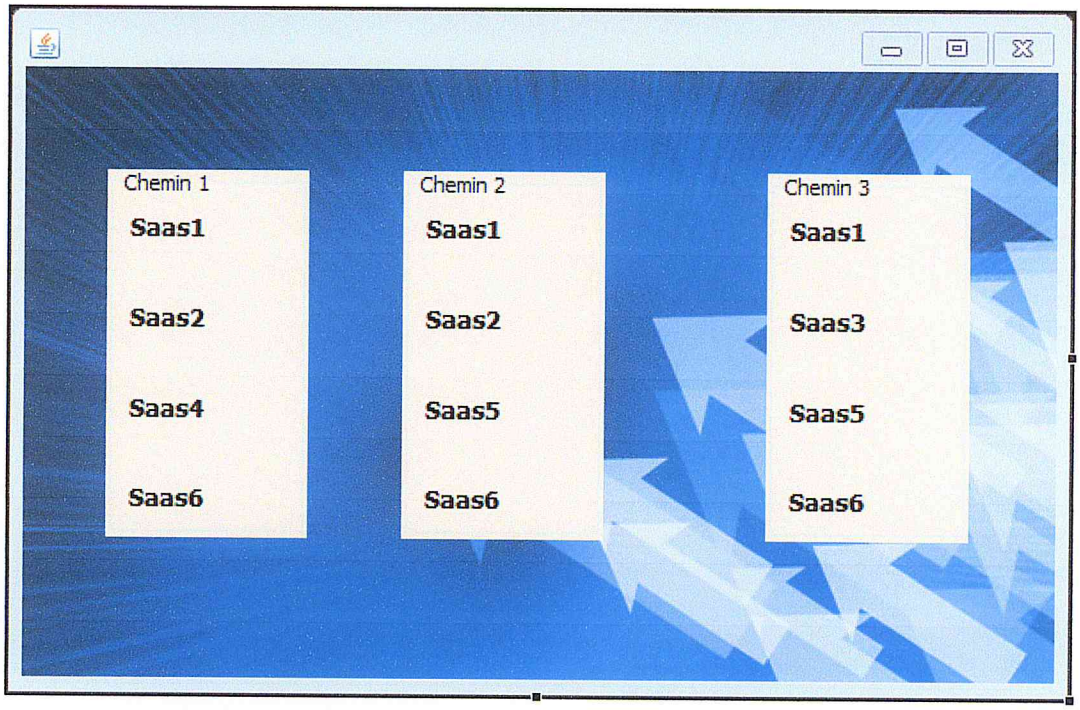


FIGURE IV.11 – *C*hemins possible

Ensuite, nous avons eu le meilleur chemin d'après les chemins obtenus dans la phase précédente en utilisant l'algorithme de Djikstra(figureIV.12).

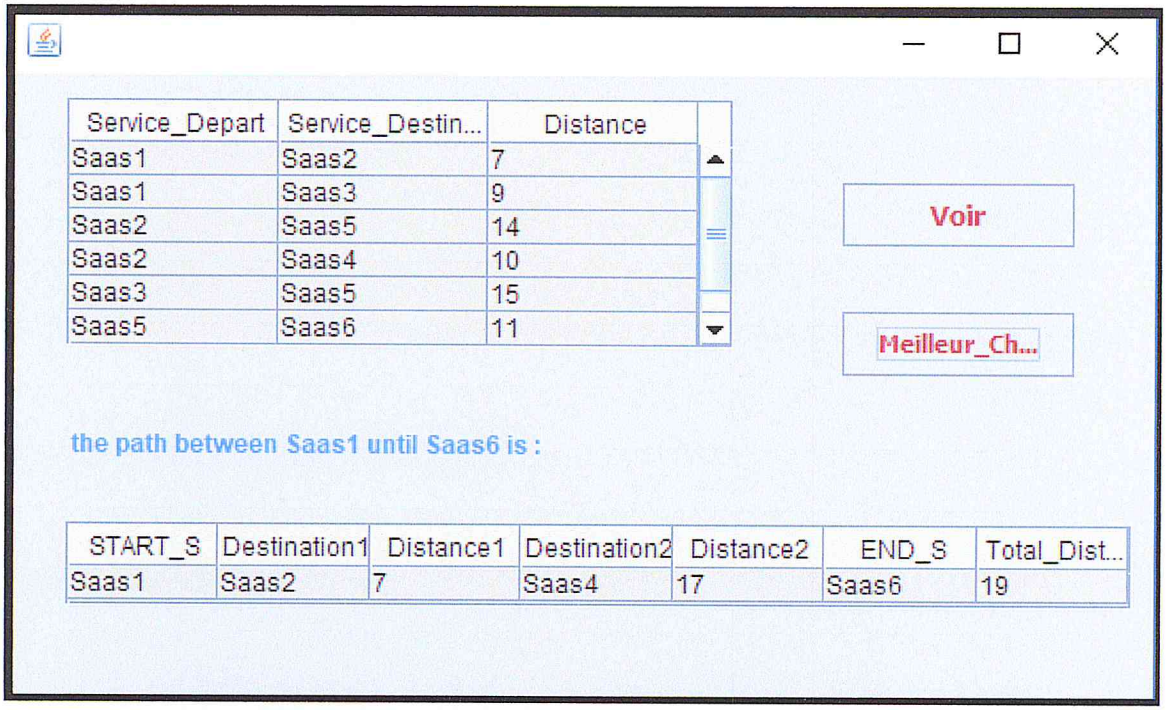


FIGURE IV.12 – *M*eilleur chemin

Finalement, pour réaliser la partie composition nous avons pris l'agence de voyages comme un exemple. Cet exemple est une mise en œuvre d'un service qu'une agence de voyages peut fournir à ses clients. Le processus BPEL permet de réserver un vol, une voiture, un hôtel pour un séjour et Le paiement est effectué par la banque.

Le processus retourne au client un message indiquant que la réservation s'est bien déroulée et ses identifiants de réservation ou bien la réservation est échouée (figureIV.13).

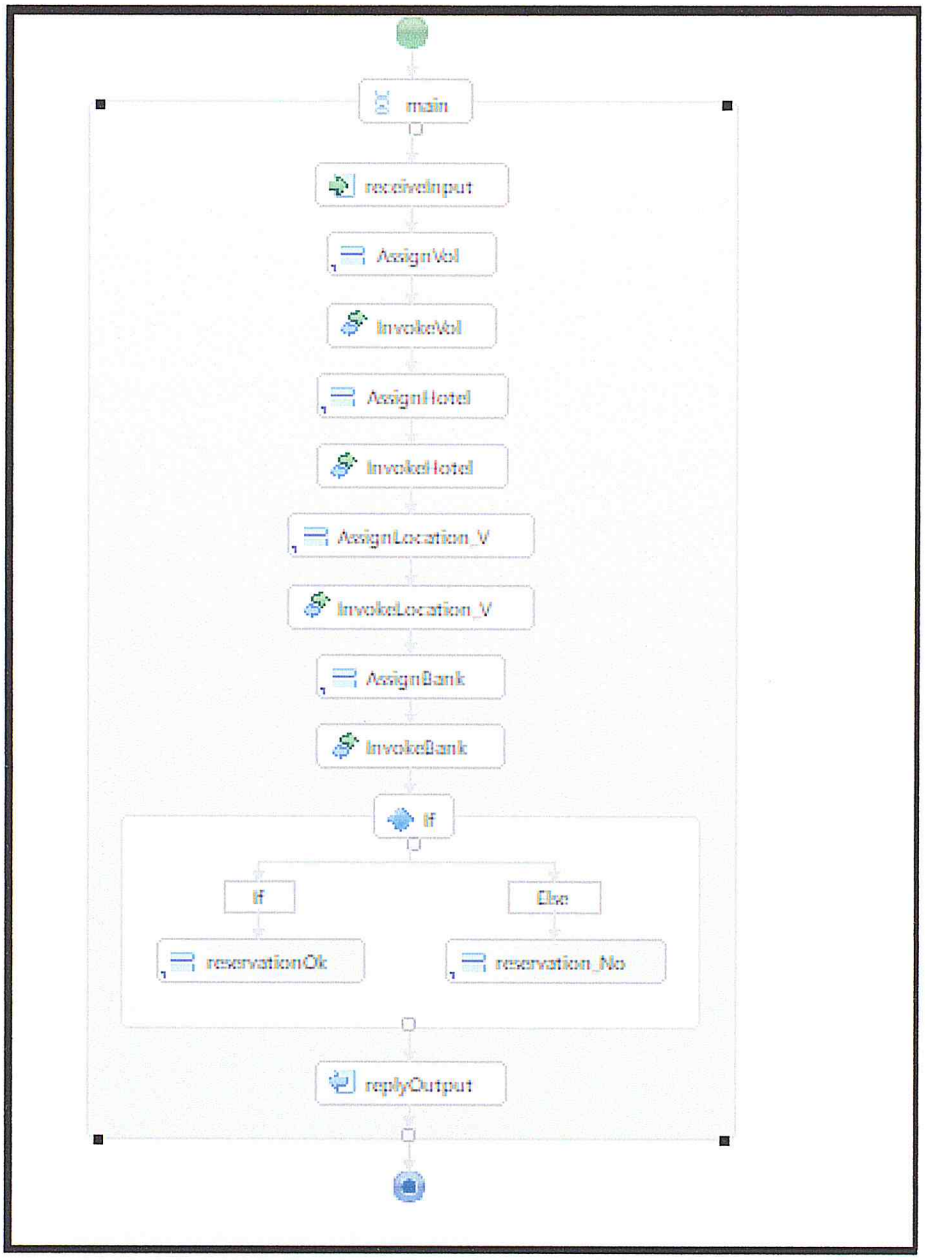


FIGURE IV.13 – *S*chéma de composition

IV.4 Conclusion :

Dans ce chapitre, nous avons illustré une expérimentation de notre solution qui été implémentée en langage JAVA en utilisant Eclipse. Nous avons essayé de mettre en oeuvre l'ensemble des idées et concepts qui caractérisent le modèle proposé.

Au premier temps, nous avons simulé les paramètres de Cloudlet correspond à un seul Cloud qui affiche un graphe mesurant le temps d'exécution en fonction des valeurs attribuées à un paramètre appartenant au Cloudlet.

Par la suite, nous avons pris en considération les règles d'interaction, l'ordonnancement des services Clouds et l'optimisation des solutions existants afin de réaliser l'étape de composition.

Conclusion Générale

Actuellement, la recherche d'un moteur de composition pour les logiciels en tant que service dans l'environnement du Cloud Computing est très important, pour répondre aux exigences des utilisateurs dans un temps minimal.

Le but de notre travail est de proposer une solution pour la composition des services Cloud de type SaaS. Pour atteindre cet objectif nous avons commencé dans un premier temps par présenter les notions de base du Cloud Computing, après nous avons étudié la composition dans les services Cloud qui nous a permis de faire une recherche bibliographique sur les travaux qui sont en relation avec notre problématique. Après cette étape, nous avons réalisé une étude comparative des solutions existantes.

L'idée de notre solution consiste à utiliser la planification de l'intelligence artificielle. Pour cela, nous avons utilisé l'algorithme Graphplan pour générer un plan valide introduisant un ensemble de règles d'interactions permettant la composition entre les services (SaaS) Clouds. En plus, l'utilisation de l'algorithme de Djikstra pour l'optimisation du plan résultant. Finalement, et après l'optimisation nous avons fait la composition entre les services Cloud en utilisant BPEL.

Pour l'expérimentation, nous avons proposé une application java permettant d'automatiser le processus de composition.

Comme perspectives de notre travail, la solution proposée peut être utilisée dans une architecture de courtage entre plusieurs fournisseurs de services Cloud et peut être utilisée aussi avec un mécanisme de sélection basé sur la qualité de services.

Bibliographie

[1] A. Sheth and A. Ranabahu, "Semantic Modeling for Cloud Computing, Part 2," IEEE Internet Computing, vol. 14, Issue : 4, pp. 81-83, 2010. DOI : 10.1109/MIC.2010.98

[2] J. McKendrick. "Does Platform as a Service have interoperability issues ?". http ://www.zdnet.com/blog/service-oriented/does-platform-as-a-service-have-interoperability-issues/4890. [Dernièreconsultation : 17/12/2017].

[3] The Economist. "Battle of the Clouds". http ://www.economist.com/opinion/displa ystory.cfm ?story.id=14644393. [Derrière consultation : 17/12/2017].

[4] D. Catteddu and G. Hogben. "Cloud Computing-Benefits, risks and recommendations for information security". http ://www.enisa.europa.eu/act/rm/files/deliverables/cloud-computing-risk-assessment/. [Derrière consultation : 17/12/2017].

[5] Moghaddam,F.,Ahmadi,M.,Sarvari,S.,Eslami,M.,Golkar,A. Telematics and Future Generation Networks (TAFGEN), 2015 1st International Conference on.IEEE.

[6] Arunkumar,G.,Neelanarayanan,V.(2015). A Novel Approach to Address Inter-operability Concern in Cloud Computing. Procedia Computer Science,Vol.50 , pp 554-559.

[7] Almubaddel,M.,Elmogy,A. Proceedings of the International Conference on Internet of things and Cloud Computing.ACM, 2016.

[8] Floerecke and Lehner. An Enhanced Cloud Ecosystem Model, la conférence européenne sur les systèmes d'information (ECIS), İstanbul, Turquie, 2016.

[9] Buyya,R.,Yeo,C.,S.,Venugopal,S.,Broberg,J.,Brandic,I.(2009). Cloud computing and emerging IT platforms : Vision, hype, and reality for delivering computing as the 5th utility.Future Generation computer systems, Vol. 25 No .6 ,pp 599-616.

[10] Rani,B,K.,Rani,B,P.,Babu,A,V.(2015). Cloud Computing and Inter-Clouds–Types, Topologies and Research Issues.Procedia Computer Science,Vol. 50 , pp 24-29.

[11] Jeff Barr, Le Cloud Computing avec Amazon Web Service, Collection : Référence, edition : Pearson France. 27 mai 2011.

[12] Mell,Peter.,Grance,Tim.,and others. The NIST Definition of Cloud Computing. National Institute of Standards and Technology, 2011.

[13] Data as a Service 101 : The Basics and Why They Matter ... .http ://www.dataversity.net/data-as-a-service-101-the-basics-and-why-they matter.[Dernière consultation : 6/12/2017].

[14] Cimpl Blog. "Qu'est-ce que le XaaS (Everything-as-a-Service) ?". http ://blog.cimpl.com/fr/quest-ce-que-le-xaas-everything-as-a-service.[Dernière consultation : 6/12/2017].

[15] Definition from Techopedia. "What is Data as a Service (DaaS) ?" . https ://www.techopedia.com/definition/28560/data-as-a-service-daas.[Dernière consultation : 6/12/2017].

[16] WhatIs TechTarget. "What is framework ?" . http ://whatis.techtarget.com/definition/framework.[Dernière consultation : 6/12/2017].

[17] https ://www.capgemini.com/consulting-fr/2011/05/le-phenomene-xaas-veritable-tsunami-informatique.[Dernière consultation : 6/12/2017].

[18] SearchSDN.com. "What is Network as a Service (NaaS) ?" . http ://searchsdn.techtarget.com/definition/Network-as-a-Service-NaaS.[Dernière consultation : 6/12/2017].

[19] itandsi. "les fondamentaux du cloud computing" https ://itandsi.files.wordpress.com/2014/09/les-fondamentaux-du-cloud-computing.pdf. [Dernier consultation 6/12/2017].

[20] Kaur,S.,Sumesh,S.,Gurminder,Kaur. Cloud Computing Interoperability : Introduction,Concerns and Challenges.Gurminder International Journal. 2017.

[21] Academia.edu. "The Ethics of Cloud Computing". http ://www.academia.edu/6348438/Le Cloud Computing[Dernière consultation : 06/12/2017]

[22] Radack,S. Cloud Computing :A Review of Features,Benefits,and Risks,and Recommendations for Secure, Efficient Implementations. National Institute of Standards and Technology, 2012.

[23] Mezga,I,n.,Rauschecker,U.(2014).The challenge of networked enterprises for cloud computing interoperability. Elsevier, Computers in Industry,Vol .65, No. 4, pp 657–674.

[24] J. R. Winkler. La sécurité dans le Cloud. Pearson Education France. 2011. https ://www.pearson.fr/resources/titles/27440100447280/extras/2509-chap01.pdf

[25] István,M.,Ursula,R.(2014). The challenge of networked enterprises for cloud computing interoperability. Computers in Industry,Vol. 165 No. 4,pp 657-674.

[26] S. Garfinkel, Architects of the information society, Edited by Harold Abelson, 1999.

[27] L. Kleinrock. A vision for the Internet. ST Journal of Research, 2(1) : 4-5, November 2005.

[28] I. Foster and C. Kesselman. The grid 2 : Blueprint for a new computing infrastructure. Morgan Kaufmann, 2003.

[29] M.A. Vouk. Cloud Computing-issues, research and implementations. Journal of Computing and Information Technology, 16(4) : 235-246, 2008.

[30] Amazon EC2. http ://aws.amazon.com/en/ec2. [Dernière consultation : 10/12/2017].

[31] Google and IBM announced University Initiative to Adress Internet-Scale Computing Challenges. http ://www-03.ibm.com/press/us/en/pressrelease/22414.wss. [Dernière consultation : 10/12/2017].

[32] Google and I.B.M join in 'Cloud Computing' Research. http ://www.nytimes.com/2007/10/08/technology/08cloud.html. [Dernière consultation : 10/12/2017].

[33] IBM Introduces Ready-To-Use Cloud Computing. http ://www-03.ibm.com/press/us/en/pressrelease/22613.wss. [Dernière consultation : 11/12/2017].

[34] Stefan Ried, Holger Kisker, Pascal Matzke, Andrex Bartels and Miroslaw Lisserman. Understanding and quantifying the future of cloud computing. Technical Report, 2011.

[35] Buyya,R., Ranjan, R., Calheiros., Rodrigo, N. "Intercloud : Utility-oriented federation of cloud computing environments for scaling of application services", In International Conference on Algorithms and Architectures for Parallel Processing. Springer,2010, pp. 13–31.

[36] Celesti, F. Tusa, M. Villari, and A. Puliafito, How to Enhance Cloud Architectures to Enable Cross-Federation, Proc. IEEE Third Int'l Conf. Cloud Computing (CLOUD), pp. 337-345, 2010.

[37] HOGAN MICHAEL, LIU FANG, SOKOL ANNIE ET TONG JIN : NIST cloud computing standards roadmap. NIST Special Publication 35, 2011.

[38] Jeffrey J,P,T., Lu,M.(2006). 'Security Modeling and Analysis of Mobile Agent Systems', SERIES IN ELECTRICAL AND COMPUTER ENGINEERING. World Scientific Publishing Company, ISBN-13 : 978-1860946349.

[39] Jula,A., Sundararajan,E., Othman,Z.(2014). Cloud computing service composition : A systematic literature review. Expert Systems with Applications, 41(8), 3809-3824.

[40] Hayyolalam, V., Kazem, A,A,P. A systematic literature review on QoS-aware service composition and selection in cloud environment. Journal of Network and Computer Applications, 2018.

[41] Vakili, A., Navimipour, N,J. Comprehensive and systematic review of the service composition mechanisms in the cloud environments. Journal of Network and Computer Applications, Vol. 81, pp 24-36. 2017.

[42] Kurdi, H., Al-Anazi, A., Campbell, C., Al Faries, A.(2015). A combinatorial optimization algorithm for multiple cloud service composition. Computers & Electrical Engineering, Vol. 42,pp 107-113.

[43] Mei, L., Chan, WK., Tse, TH. A tale of clouds : paradigm comparisons and some thoughts on research issues. In : Proc IEEE Asia–Pacific services computing conference. IEEE Computer Society, Washington, pp 464–469, 2008.

[44] Qu, L. ,others. "Credible service selection in cloud environments", In Sydney, Australia : Macquarie University.(2016).

[45] Liu, Y., Esseghir, M., Boulahia, L, M.(2016). Evaluation of parameters importance in cloud service selection using rough sets. Applied Mathematics,Vol. 7, No. 6, pp 527

[46] Huo, Y., Zhuang, Yi., Gu, J., Ni, S., Xue, Yu.(2015). Discrete gbest-guided artificial bee colony algorithm for cloud service composition. Applied Intelligence, Vol. 42, No. 4, pp 661-678.

[47] Hu,J., Guo, C., Wang, H., Zou, P.(2005). Quality driven web services selection. In : Proceeding of the IEEE International Conference on e-Business Engineering, Beijing,China, pp 681-688, 2005.

[48] Alrifai, M., Risse, T. Combining global optimization with local selection for efficient QoS-aware service composition. In : Proceeding of the 18th International Conference on World Wide Web Madrid, Spain, pp 881-890, 2009.

[49] Zeng, L., Benatallah, B., Ngu, A., Dumas, M., Kalagnanam, J., Chang, H.(2004). QoS-aware middleware for web services composition. IEEE Trans Software Eng ,Vol. 30, No. 5, pp 311–327.

[50] Ardagna, D., Pernici, B.(2007). Adaptive service composition in flexible processes, IEEE Trans Software Eng, Vol. 33, No. 6, pp 369–384.

[51] Ma, Y., Zhang, C.(2008). Quick convergence of genetic algorithm for QoS-driven web service selection, Comput Netw, Vol. 52, No. 5, pp 1093–1104.

[52] Gao, H., Yan, J., Mu, Y.(2014). Trust oriented QoS aware composite service selection based on genetic algorithms, Concurrency and Computation. Pract Experience Vol. 26, No. 2,pp 500–515.

[53] Wang, S., Zhu, X., Yang, F.(2014). Efficient QoS management for QoS–aware web service composition. Int J Web Grid Serv, Vol. 10, No. 1, pp 1–23.

[54] Tao, F., Zhao, D., Hu, Y., Zhou, Z.(2008). Resource service composition and its optimal-selection based on particle swarm optimization in manufacturing grid system. IEEE Trans Ind Inform, Vol. 4, No. 4, pp 315–327.

[55] Wu, Q., Zhu, Q.(2013). Transactional and QoS-aware dynamic service composition based on ant colony optimization. Futur Gener Comput Syst, Vol. 29, No. 5, pp 1112–1119.

[56] Asghari, S., Navimipour, Nima, J. Service composition mechanisms in the multi-cloud environments : a survey. Journal of Int J New Comput Archit Appl (IJNCAA), Vol. 6,pp 40-48, 2016.

[57] Chouchani, I.(2010). "Utilisation d'un algorithme génétique pour la composition de services Web", In Université du Québec à Montréal. 2010

[58] Kiran,T, R., Roshini,G., Harshini,K, S., Gayathri,G, A. "An Empirical Model of Data Integrity in Multi Cloud Data Storage."

[59] Zou, G., Chen, Y., Xiang, Y., Huang, R., Xu, Y. AI planning and combinatorial optimization for web service composition in cloud computing. In : Proceedings of the international conference on cloud computing and virtualization, pp 1–8, 2010.

[60] Jula, A., Othman, Z., Sundararajan, E.(2015). Imperialist competitive algorithm with PROCLUS classifier for service time optimization in cloud computing service composition. Expert Systems with Applications, Vol. 42, No. 1, 135-145.

[61] Wang, X., Cao, J., Xiang, Y. Dynamic cloud service selection using an adaptive learning mechanism in multi-cloud computing. Journal of systems and Software, Vol. 100, pp 195-210, 2015.

[62] Gutierrez-Garcia, J, O., Sim, K, M.(2013). Agent-based cloud service composition. Applied intelligence, Vol. 38, No. 3, pp 436-464.

[63] Kholidy, H, A., Hassan, H., Sarhan, A, M., Erradi, A., Abdelwahed, S.(2015). QoS Optimization for Cloud Service Composition Based on Economic Model Internet of Things. User-Centric IoT , pp 355-366 :Springer.

[64] Karimi, M, B., Isazadeh, A., Rahmani, A, M. QoS-aware service composition in cloud computing using data mining techniques and genetic algorithm. The Journal of Supercomputing, pp 1-29, 2016.

[65] Techno-Science.net. "Extreme programming". http ://www.technoscience.net/ ?onglet=glossaire definition=723 . [Dernière consultation : 1/08/2018].

[66] Airbrake. "Extreme Programming : What Is It And How Do You Use It ?" . https ://airbrake.io/blog/sdlc/extreme-programming. [Dernière consultation : 1/08/2018].

[67] Piloter.org. "XP eXtreme Programming, les méthodes agiles". https ://www.piloter.org/projet/methode/xp.htm . [Dernière consultation : 1/08/2018].

[68] "Extreme Programming : A Gentle Introduction" . http ://www.extremeprogramming.org/. [Dernière consultation : 1/08/2018].

[69] Berners-Lee., Tim., Hendler., James.,Lassila., Ora. The semantic web. The journal of Scientific american, pp 34-43, 2001.

[70] Oussama, KAMEL. "DECOUVERTE ET COMPOSITION DE WEB SERVICES SEMANTIQUES", Université Mohamed Boudiaf de M'sila, 2012.

[71] Quang, P., Tien., T. "Ontologies et web services". The journal of Activity report. Institut de la Francophonie pour l'Informatique, 2005.

[72] Howe, A et all., "PDDL — The Planning Domain Definition Language". AIPS-98 Planning Competition Committee, UCPOP language manual, University of Washington.

[73] Vernhes, S. "Décomposition des problèmes de planification de tâches basée sur les landmarks". UNIVERSITE DE TOULOUSE, 2014.

[74] Najjar, M, M. "Planification par des ordres partiels". 2e cycle, Département des mathématiques et de l'informatique, faculté des sciences, université Sherbrooke, Canada, 2000.

[75] Evans, J. "Optimization algorithms for networks and graphs", In Routledge. 2017.

[76] Oracle. "BPEL Process Manager". https ://www.oracle.com/technetwork/middleware/bpe /overview/index.html. [Dernière consultation : 1/08/2018].

[77] Silkhom. "Le langage Java : histoire, caractéristiques et popularité". https ://www.silkhom.com/langage-java-histoire-caracteristiques-popularite/. [Dernière consultation : 20/08/2018].

[78] Wikilivres. "Programmation Java/Introduction" . https ://fr.wikibooks.org/wiki/Programmation-Java/Introduction. [Dernière consultation : 20/08/2018].

[79] Eclipse – Un environnement de développement libre, extensible et ... https ://www.opensourcemacsoftware.org/developpement/eclipse-ide-java-mac.html. [Dernière consultation : 20/08/2018].

[80] Avrim L. Blum et Merrick L. Furst. Fast Planning Through Planning Graph Analysis. Final version in artificial Intelligence, 90 :281-300, 1997.

[81] J. Peer, "Web Service Composition as AI Planning - a Survey," University of St. Gallen, Switzerland, Tech. Rep., 2005.

[82] A. Blum and M. Furst, "Fast Planning Through Planning Graph Analysis," in Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 95), 1995, pp. 1636–1642.

[83] PairForm. "Modèles et Algorithmes de Graphe - Algorithme de Dijkstra". https ://www.pairform.fr/doc/1/32/180/web/co/Dijkstra.html. [Dernière consultation : 15/09/2018].

[84] Interstices. "Le plus court chemin". https ://interstices.info/le-plus-court-chemin/. [Dernière consultation : 15/09/2018].

[85] LANANI SADOK, «Une approche BPM (Business Process Managment) par composition d'applications dans le Cloud Computing», Mémoire de magister, UNIVERSITE MOHAMED KHIDER, BISKRA.

[86] The CloudSim Framework : Modelling and Simulating the Cloud. https ://opensourceforu.com/2014/03/cloudsim-framework-modelling-simulating-cloud-environment/. [Dernier consultation 20.08.2018]

[87] M. C. Lab, "Cloudsim : A framework for modeling and simulation of cloud computing infrastructures and services.https ://code.google.com/archive/p/cloudsim/. [Dernier consultation 20/08/2018]

[88] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose, and R. Buyya, "Cloudsim : A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, "Software :Practice and Experience (SPE), vol. 41, no. 1, pp. 23–50, 2011.

[89] SearchCloudComputing. "What is cloudlet ? Definition from WhatIs.com". https ://searchcloudcomputing.techtarget.com/definition/cloudlet. [Dernier consultation 20/08/2018]

Annexes

Annexe A. Fichier WSDL des Saas

<?xml version='1.0' encoding='UTF-8'?><rdf :RDFxmlns :owl = "http ://www.w3.org/2002/07/owl#"
xmlns :rdfs = "http ://www.w3.org/2000/01/rdf-schema#" xmlns :rdf = "http ://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns :service = "http ://www.daml.org/services/owl-s/1.1/Service.owl#" xmlns :process = "http ://www.daml.org/services/owl-s/1.1/Process.owl#" xmlns :profile = "http ://www.daml.org/services/owl-s/1.1/Profile.owl#" xmlns :grounding = "http ://www.daml.org/services/owl-s/1.1/Grounding.owl#" xmlns :expr = "http ://www.daml.org/services/owl-s/1.1/generic/Expression.owl#" xmlns :swrl = "http ://www.w3.org/2003/11/swrl#" xml :base = "http ://127.0.0.1/services/My-Composition.owl">
<owl :Ontologyrdf :about=""><owl :importsrdf :resource="http ://127.0.0.1/ontology/Ontologie.owl" / ></owl :Ontology>
<service :Servicerdf :ID="My-Composition">
<service :presentsrdf :resource="My-Profile" / >
<service :describedByrdf :resource="CompositeProcess-1" / >
<service :supportsrdf :resource="CompositionGrounding" / >
< /service :Service>
<profile :Profilerdf :ID="My-Profile">
<service :isPresentedByrdf :resource="My-Composition-Service" / >
<profile :serviceNamexml :lang="en" ¿My-Composition¡/profile :serviceName>
<profile :textDescriptionxml :lang="en">
Composition des services cloud de type Saas. < /profile :textDescription>
<profile :hasInputrdf :resource="#InSaas1" / >
<profile :hasInputrdf :resource="#InSaas2" / >
<profile :hasInputrdf :resource="#InSaas3" / >
<profile :hasInputrdf :resource="#InSaas4" / >
<profile :hasInputrdf :resource="#InSaas5" / >
<profile :hasInputrdf :resource="#InSaas6" / >
<profile :hasInputrdf :resource="#InSaas7" / >
<profile :hasInputrdf :resource="#InSaas8" / >
<profile :hasOutputrdf :resource="#OutSaas1" / >
<profile :hasOutputrdf :resource="#OutSaas2" / >
<profile :hasOutputrdf :resource="#OutSaas3" / >
<profile :hasOutputrdf :resource="#OutSaas4" / >
<profile :hasOutputrdf :resource="#OutSaas5" / >
<profile :hasOutputrdf :resource="#OutSaas6" / >
<profile :hasOutputrdf :resource="#OutSaas7" / >
<profile :hasOutputrdf :resource="#OutSaas8" / >
< /profile :Profile>
¡process :ProcessModelrdf :ID="My-Composition-Process-Model"¿ ¡service :describesrdf :resource="#My-Composition-Service" / >
¡process :hasProcessrdf :resource="#My-Composition-Process" / >
< /process :ProcessModel¿

<process :AtomicProcessrdf :ID="ExecuterServices"¿ <process :hasInputrdf :resource="#InSaas3" / >
<process :hasInputrdf :resource="#InSaas4" / >
<process :hasInputrdf :resource="#InSaas5" / >
<process :hasOutputrdf :resource="OutSaas3" / >
<process :hasOutputrdf :resource="OutSaas4" / >
<process :hasOutputrdf :resource="OutSaas5" / >
< /process :AtomicProcess> <process :Inputrdf :ID="InSaas3" > <process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#any
URI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud1<
/process :parameterType>
<rdfs :label>InSaas3< /rdfs :label>
< /process :Input>
<process :Inputrdf :ID="InSaas4" >
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#any
URI">http ://127.0.0.1/ontology/ontologie.owl#Cloud2<
/process :parameterType>
<rdfs :label>InSaas4< /rdfs :label>
< /process :Input>
<process :Inputrdf :ID="InSaas5" >
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#any
URI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud2<
/process :parameterType>
<rdfs :label>InSaas5< /rdfs :label>
< /process :Input>
<process :Outputrdf :ID="OutSaas3" >
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#any
URI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud1<
/process :parameterType>
<rdfs :label>OutSaas3< /rdfs :label>
< /process :Output>
<process :Outputrdf :ID="OutSaas4" >
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#any
URI"¿http :127.0.0.1/ontology/Ontologie.owl#Cloud2¡/process :parameterType>
<rdfs :label>OutSaas4< /rdfs :label>
< /process :Output>
<process :Outputrdf :ID="OutSaas5" >
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#any
URI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud2<
/process :parameterType>
<rdfs :label>OutSaas5< /rdfs :label>
< /process :Output>
<process :AtomicProcessrdf :ID="DemandeServices1" >

<process :hasInputrdf :resource="#OutSaas3" / >
<process :hasInputrdf :resource="#OutSaas4" / >
<process :hasInputrdf :resource="#OutSaas5" / >
<process :hasOutputrdf :resource="OutSaas8" / >
<process :hasOutputrdf :resource="OutSaas6" / >
<process :hasOutputrdf :resource="OutSaas2" / >
< /process :AtomicProcess¿ <process :Inputrdf :ID="OutSaas3"¿ <process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#anyURI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud1¡/process :parameterType¿
<rdfs :label>OutSaas3< /rdfs :label>
< /process :Input>
<process :Inputrdf :ID="OutSaas4">
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#anyURI">http ://127.0.0.1/ontology/ontologie.owl#Cloud2<
/process :parameterType>
<rdfs :label>OutSaas4< /rdfs :label>
< /process :Input>
<process :Inputrdf :ID="OutSaas5">
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#anyURI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud2<
/process :parameterType>
<rdfs :label>OutSaas5< /rdfs :label> < /process :Input> <process :Outputrdf :ID="OutSaas8"> <process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#anyURI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud3<
/process :parameterType>
<rdfs :label>OutSaas3< /rdfs :label>
< /process :Output>
<process :Outputrdf :ID="OutSaas6">
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#anyURI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud3<
/process :parameterType>
<rdfs :label>OutSaas6< /rdfs :label>
< /process :Output>
<process :Outputrdf :ID="OutSaas2">
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#anyURI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud1<
/process :parameterType>
<rdfs :label>OutSaas2< /rdfs :label>
< /process :Output>
<process :AtomicProcessrdf :ID="DemandeServices2">
<process :hasInputrdf :resource="#OutSaas6" / >
<process :hasOutputrdf :resource="#OutSaas1" / >
< /process :AtomicProcess> <process :Inputrdf :ID="OutSaas6">
<process :parameterType rdf :datatype="http ://www.w3.org/2001/XMLSchema
#anyURI">http ://127.0.0.1/ontology/Ontologie.owl#Cloud3¡/process :parameter
Type>

<rdfs :label¿OutSaas6¡/rdfs :label>
< /process :Input>
<process :Outputrdf :ID=”OutSaas1”>
<process :parameterType rdf :datatype=”http ://www.w3.org/2001/XMLSchema
#anyURI”>http ://127.0.0.1/ontology/Ontologie.owl#Cloud1¡/process :para-
meter
Type>
<rdfs :label¿OutSaas1¡/rdfs :label>
< /process :Output>
<process :AtomicProcessrdf :ID=”DemandeServices3”>
<process :hasInputrdf :resource=”#OutSaas1” / >
<process :hasOutputrdf :resource=”#OutSaas7” / >
< /process :AtomicProcess>
<process :Inputrdf :ID=”OutSaas1”>
<process :parameterType rdf :datatype=”http ://www.w3.org/2001/XMLSchema
#anyURI”>http ://127.0.0.1/ontology/Ontologie.owl#Cloud1<
/process :parameterType/ >
<rdfs :label>OutSaas1< /rdfs :label>
< /process :Input>
<process :Outputrdf :ID=”OutSaas7”>
<process :parameterType rdf :datatype=”http ://www.w3.org/2001/XMLSchema
#anyURI”>http ://127.0.0.1/ontology/Ontologie.owl#Cloud3<
/process :parameterType>
<rdfs :label>OutSaas7< /rdfs :label>
< /process :Output>
<process : CompositeProcessrdf : ID=CompositeProcess-1>
<process : composedOf>
<process : Sequence>
<process : components>
<process : ControlConstructList>
<list : first >
<process : Performrdf : ID=PerformExecuterServices>
<process : processrdf : resource=#ExecuterServices/ >
< /process : Perform>
< / list : first>
<list : rest>
<process : ControlConstructList>
<list : first >
<process : Perform>
<process : processrdf : resource=#PerformDemandeServices1/ >
<process : hasDataFrom>
<process : Binding>
<process : toParamrdf : resource=#OutSaas3/ >
<process : toParamrdf : resource=#OutSaas4/ >
<process : toParamrdf : resource=#OutSaas5/ >
<process : valueSource>

```
<process : valueOf>
<process : theVarrdf : resource=#OutSaas3/ >
<process : theVarrdf : resource=#OutSaas4/ >
<process : theVarrdf : resource=#OutSaas5/ >
<process : fromProcessrdf : resource=Process . owl#TheParentPerform/ >
<list : rest>
<process : ControlConstructList>
<list : first >
<process : Perform>
<process : processrdf : resource=#PerformDemandeServices2/ >
<process : hasDataFrom>
<process : Binding>
<process : valueSource>
<process : valueOf>
<process : theVarrdf : resource=#OutSaas6/ >
<process : fromProcessrdf : resource=Process . owl#TheParentPerform/ >
<list : rest>
<process : ControlConstructList>
<list : first >
<process : Perform>
<process : processrdf : resource=#PerformDemandeServices3/ >
<process : hasDataFrom>
<process : Binding>
<process : toParamrdf : resource=#OutSaas1/ >
<process : valueSource>
<process : valueOf>
<process : theVarrdf : resource=#OutSaas1/ >
<process : fromProcessrdf : resource=Process . owl#TheParentPerform/ >
<grounding :WsdlGroundingrdf :ID="CompositionGrounding">
<service :supportedByrdf :resource="#My-Composition-Service"/ >
< /grounding :WsdlGrounding>
< /rdf :RDF>
```

Annexe B. Planificateur GraphPlan

1. GraphPlan :

Le planificateur Graphplan proposé par A. Blum et M. Furst[80] génère un graphe de synthèse à partir des conditions initiales en appliquant les actions de base dont il dispose. A partir du graphe, il essaye en appliquant certaines règles de trouver une séquence d'actions menant aux objectifs prédéfinis.

1.1. Le graphe de synthèse :

Le graphe de synthèse construit par Graphplan est un graphe par niveau constitué de deux types de noeuds et de trois types d'arcs. Les niveaux du graphe sont de deux sortes : des niveaux de propositions alternés avec des niveaux d'actions. Les niveaux de propositions contiennent les noeuds de propositions, alors que les niveaux d'actions sont formés par les noeuds d'actions[74].

Le premier niveau du graphe est un niveau de propositions ou chaque noeud y figurant représente une proposition existante parmi les conditions initiales. Un graphe de synthèse est résumé comme suit : des propositions vraies a un temps (t), des actions possiblement applicables a un temps (t), des propositions possiblement vraies a un temps (t+1), des actions possiblement applicables a un temps (t+1), des propositions vraies a un temps (t+2), et ainsi de suite[74].

Les arcs dans le graphe schématisent les relations entre les actions et les propositions. Un noeud d'action dans un niveau d'action (i) est connecté par : (1) les arcs de préconditions à ses noeuds de préconditions situées dans le niveau de propositions (i) , (2) des arcs d'effets ajoutés à ses noeuds de post-conditions qui sont dans le niveau de propositions (i+1), (3) des arcs d'effets supprimés à ses effets supprimés dans le niveau de proposition (i+1)[74].

1.2. Description de l'algorithme :

L'algorithme général de la création du graphe est décrit dans la figure ci dessous. la création du graphe se fait par couches, une couche (i) est composée d'un niveau (i) d'actions et d'un niveau (i+1) de propositions[74].

Au début le premier niveau est un niveau de propositions représentant les conditions initials (P0) ainsi d'une liste des actions prédéfinis (A) et un indice (i). Ensuite, et d'aprés la (ligne 3 jusqu'à la ligne 6) on construit l'ensemble des actions(Ai+1) où ces conditions sont dans le niveau de proposition(i), ainsi la prochaine proposition(Pi+1) c'est l'union de proposition de niveau(i) avec l'effet de l'action(Ai+1). La boucle 'for each' permet de faire des liens entres l'action avec sa précondition(ligne 8), et liens entre l'action et son effet(ligne 9). dans la (ligne 12) on passe d'un niveau (i) au prochaine niveau (i+1). la condition 'if'(ligne 14) permet de tester si le niveau de proposition(i) et (i-1) sont identiques alors il n'y a pas une solution Graphplan.

Input : A /*action set*/
P0 /*initial state*/

Output : GP /* GP graph planning */

BEGIN

1 : GP $\leftarrow$ P0

2 : i $\leftarrow$ 0

3 : repeat
4 : Ai+1 $\leftarrow$ a/ a $\in$ A$\wedge$ Pre (a) $\subseteq$ Pi /*set of action where these conditions are in Pi*/

5 : Pi+1 $\leftarrow$ Pi $\cup$ Eff(a) / a $\in$ Ai+1 /*the next proposition is the union of Pi with the effect of the action Ai+1*/

6 : GP $\leftarrow$ GP $\cup$ Ai+1 $\cup$ Pi+1

7 : for each a $\in$ Ai+1

8 : link a with Precondition arcs to Pre (a) in Pi

9 : link a with Effect arcs to Eff(a) inPi+1

10 : mak no-op

11 : end for

12 : i $\leftarrow$ i+1

13 : if Pi= Pi$-$1 then

14 : write NO-GRAPH-SOLUTION

15 : end if

16 : return GP

END

Algorithme de création de graphe

1.3. Complexité de l'algorithme :

Le temps pris par l'algorithme pour créer un graphe de synthèse est polynomiale en considèrent la longueur de la description du problème et le nombre des

étapes.

Théorème : Considérons un problème de planification avec n objets, p propositions dans les conditions initiales, et m STRIPS opérant chacun avec un nombre constant de paramètres formels. Soit l la longueur de la liste d'ajout la plus longue de tous les opérateurs. Ainsi, la taille d'un graphique de planification en t créé par Graphplan et le temps nécessaire pour créer le graphique sont polynomiales n, m, p, l et t[80].

Soit k le plus grand nombre de paramètres formels dans n'importe quel opérateur. Étant donné que les opérateurs ne peuvent pas créer de nouveaux objets, le nombre de propositions différentes qui peuvent être créées par l'instanciation d'un opérateur est O (lnk). Ainsi, le nombre maximal de noeuds dans n'importe quel niveau de proposition du graphe de planification est O (p + mlnk).

Étant donné que tout opérateur peut être instancié en O (nk) dans différent cas possible, le nombre maximal de noeuds dans n'importe quel niveau d'action de planification graphique est O (mnk). De ce fait, la complexité du graphe de planification est polynomiale dans n, m, p, l et t, puisque k est constant[80].

1.4. Avantages d'utilisation de graphplan :

Les avantages que procure Graphplan afin de permettre sa réalisation peuvent se résumer comme suit :

- Graphplan réalise un gain énorme en temps en décidant de construire le graphe de synthèse à la première phase avant la recherche de plan. Une fois le graphe créé, il n'a pas besoin d'instancier les actions à partir des opérateurs au cours de la planification, car toutes les actions dont il pourrait avoir besoin ont été formulées au moment de la création du graphe[74].

- Graphplan a de bonnes performances. Le travail majeur dans Graphplan est la création du graphique de planification. La complexité de la création d'un graphique de planification est d'ordre polynômial faible en considérant le nombre d'actions et de propositions, la dépense du temps et de l'espace dans Graphplan est plutôt Plus réduite par rapport à la majorité des planificateurs IA[81].

- Graphplan s'arrêtera automatiquement, quand aucun plan valide n'existe. C'est-à-dire, lorsque deux couches propositionnelles consécutives sont identiques, la construction du Graphique de planification s'arrête[82].

Annexe C. Algorithme de Djikstra

1. Lgorithme de Djikstra :

Cet algorithme est dû à Edsger W. Dijkstra. C'est l'un des algorithmes qui permet de trouver le plus court chemin d'un sommet donné vers tous les autres sommets. Il n'est utilisable que pour les graphes qui n'ont pas de circuit de coût négatif[83].

1.1. Description de l'algorithme :

Input : G =(X,A) /*G the Solution's Graph*/
W /* edges weight*/
s ∈ X
Output : V , P /* V table stores the vertex labels of G */ P table allowing to find the composition of the paths*/

1 : Initialize V at $+\infty$

2 : V[s]= 0 /*label s initialized to 0*/

3 : Initialize P at 0

4 : P[s]= s /*put s in the composition table p*/

5 : repeat /* look for an unsecured x top (not chosen) of smaller label */

6 : Vmin = $+\infty$

7 : For y from 1 to N do

8 : if y not marked and V[y] ¡ Vmin then

9 : x ← y

10 : Vmin ← V[y]/*
11 : if Vmin¡$+\infty$ then /* update of the unspecified successors of x*/

marked x

For any successor y of x do

if y not marked and V[x] + W[x,y] ¡ V[y] then

V[y] = V[x]+W[x,y]

P[y] = x
Until Vmin = $+\infty$

Algorithme Djikstra

1.2. Complexité de l'algorithme :

La complexité est dans le temps de parcourt dans le graphe pour détermine le plus court chemin. S'il y a n sommets, l'algorithme nécessite au plus (n - 1) étapes. À l'étape k, pour le sommet choisi x, il faut examiner ses successeurs non calculés (leur nombre est au plus égal au nombre d'arcs issus de x). Globalement,

il y a donc au plus $m + n^2$ opérations, où m est le nombre d'arcs du graphe. Comme $m \leq n^2$, le nombre d'opérations est proportionnel au carré du nombre de sommets[84].

Annexe D. CloudSim

D.1 Définition

CloudSim est un cadre de modélisation et de simulation transparentes des performances des applications, d'infrastructures et de services de Cloud Computing[85]. C'est un outil de simulation qui permet aux développeurs de Cloud de tester gratuitement les performances de leurs règles d'approvisionnement dans un environnement répétable et contrôlable[86].

Contrairement à d'autres outils de simulation (par exemple SimGrid, GangSim), CloudSim supporte[87] :

- La modélisation et la simulation des Data Centers à grande échelle dans les Cloud Computing.

- La modélisation et la simulation des serveurs virtuels avec des politiques permettant le provisionnement des ressources de la machine physique en machine virtuelle.

- La modélisation et la simulation des topologies de réseau et des applications de transmission de messages.

- La modélisation et la simulation des clouds différés.

- Supporte l'insertion dynamique des éléments de simulation, l'arrêt et la reprise de la simulation.

- Supporte les politiques définies par les utilisateurs pour l'allocation des VMs dans des hôtes et l'allocation des ressources de l'hôte dans des VMs.

- CloudSim a été étendu pour permettre la simulation des approches permettant la réduction de la consommation d'énergie des Data Centers. Ainsi la possibilité de simuler des applications de service avec une charge de travail dynamique a été incorporée.

D.2 Les classe de CloudSim

Le simulateur CloudSim est composé de plusieurs classes tels que :

▷ **Cloudlet :** Cette classe modélise les services d'application dans Cloud(tels que : la livraison, la gestion des réseaux sociaux, et le déroulement des opérations métier de l'entreprise)[88]. Le Cloudlet a été spécifiquement conçu pour prendre en charge les applications mobiles pour la reconnaissance vocale, le traitement du langage, l'apprentissage automatique et la réalité virtuelle. Son objectif est d'augmenter le temps de réponse des applications s'exécutant sur des appareils mobiles

en utilisant une réseau sans fil à faible latence et à large bande passante[89].

▷ **Datacenter :** La classe Datacenter modélise l'infrastructure de base(matériel) offert par les fournisseurs dans un environnement de Cloud Computing. Il encapsule un ensemble de machines de calcul qui peuvent être homogène ou hétérogènes dans leurs configurations de ressources (mémoire, noyau, capacité et stockage). De plus, chaque composant de Datacenter référence un composant généralisé de ressource qui implémente un ensemble d'allocation de bande passante, de mémoire et des espaces de stockage[88].

▷ **DataCentreBroker :** Cette classe modélise le courtier (Broker), qui est responsable de la médiation entre les utilisateurs et les prestataires de services selon les conditions de QoS des utilisateurs, de plus il déploie des tâches de service à travers les Clouds[88].

Le nom Broker produit sur les prestataires de services du Cloud par le service d'information du Cloud CIS (Cloud information services) en discutant avec eux pour une allocation des Confiscations qui répond aux besoins de QoS des utilisateurs[88].

▷ **Machine Virtuelle :** Cette classe modélise une instance de machine virtuelle (VM), qui est géré par le composant host Cloud, plusieurs VMs peuvent instancier un seul host et affecter des politiques prédéfinies de partage de processeur (espace partagé, temps partagé)[88].

Annexe E. WSDL des services de composition

E.1 Service-Vol

```
<?xmlversion="1.0"? >
<definitionsname="Vol"
targetNamespace="http ://vol.localhost"
xmlns :tns="http ://vol.localhost"
xmlns :plnk="http ://docs.oasis-open.org/wsbpel/2.0/plnktype"
xmlns="http ://schemas.xmlsoap.org/wsdl/"
xmlns :soap="http ://schemas.xmlsoap.org/wsdl/soap/" >

<types>
<schemaattributeFormDefault="unqualified"elementFormDefault="qualified"
targetNamespace="http ://vol.localhost"
xmlns=http ://www.w3.org/2001/XMLSchema>

<elementname="VolRequest">
<complexType>
<sequence>
<elementname="Code-V"type="string"/ >
<elementname="NB-Place"type="string">< /element>
<elementname="Class-V"type="string">< /element>
<elementname="Prix-V"type="double">< /element>
<elementname="Date-D-Vol"type="date">< /element>
< /sequence>
< /complexType>
< /element>
<elementname="VolResponse">
<complexType>
<sequence>
<elementname="V-Depart"type="string"/ >
<elementname="V-Arrivée"type="string">< /element>
<elementname="Date-Dprt"type="date">< /element>
<elementname="Date-Arrv"type="date">< /element>
<elementname="NB-Pers"type="string">< /element>
<elementname="Vol"type="boolean">< /element>
< /sequence>
< /complexType>
< /element>
< /schema>
< /types>
```

```
<messagename="VolRequestMessage">
<partname="payload"element="tns :VolRequest"/ >
< /message>
<messagename="VolResponseMessage">
<partname="payload"element="tns :VolResponse"/ >
< /message>
<portTypename="Vol">
<operationname="process">
<inputmessage="tns :VolRequestMessage"/ >
<outputmessage="tns :VolResponseMessage"/ >
< /operation>
< /portType>

<plnk :partnerLinkTypename="Vol">
<plnk :rolename="VolProvider"portType="tns :Vol"/ >
< /plnk :partnerLinkType> <bindingname="VolBinding"type="tns :Vol">
<soap :bindingstyle="document"
transport=http ://schemas.xmlsoap.org/soap/http/ >
<operationname="process">
<soap :operation
soapAction="http ://vol.localhost/process"/¿
<input>
<soap :bodyuse="literal"/ >
< /input>
<output>
<soap :bodyuse="literal"/ >
< /output>
< /operation>
< /binding>

<servicename="VolService">
<portname="VolPort"binding="tns :VolBinding">
<soap :addresslocation=http ://localhost :8080/Vol/ >
< /port>
< /service>
< /definitions>
```

E.2 Service-Hôtel

```
<?xmlversion="1.0"? >
<definitionsname="Hotel"
targetNamespace=http ://hotel.localhost
xmlns :tns=http ://hotel.localhost
xmlns :plnk=http ://docs.oasis-open.org/wsbpel/2.0/plnktype
xmlns=http ://schemas.xmlsoap.org/wsdl/
xmlns :soap=http ://schemas.xmlsoap.org/wsdl/soap/ >

<types>
<schemaattributeFormDefault="unqualified"elementFormDefault="qualified"
targetNamespace=http ://hotel.localhost-
xmlns="http ://www.w3.org/2001/XMLSchema">

<elementname="HotelRequest">
<complexType>
<sequence>
<elementname="NomH"type="string"/ >
<elementname="AddrH"type="string">< /element>
<elementname="PrixH"type="double">< /element>
< /sequence>
< /complexType>
< /element>
<elementname="HotelResponse">
<complexType>
<sequence>
<elementname="Ville"type="string"/ >
<elementname="Date-Entrée"type="date">< /element>
<elementname="Date-Sortie"type="date">< /element>
<elementname="Nbr-Personne"type="double">< /element>
< /sequence>
< /complexType>
< /element>
< /schema>
< /types>

<messagename="HotelRequestMessage">
<partname="payload"element="tns :HotelRequest"/ >
< /message>
<messagename="HotelResponseMessage">
<partname="payload"element="tns :HotelResponse"/ >
< /message>
```

```
<portTypename=”Hotel”>
<operationname=”process”>
<inputmessage=”tns :HotelRequestMessage”/ >
<outputmessage=”tns :HotelResponseMessage”/ >
< /operation>
< /portType>
<plnk :partnerLinkTypename=”Hotel”>
<plnk :rolename=”HotelProvider”portType=”tns :Hotel”/ >
< /plnk :partnerLinkType>

<bindingname=”HotelBinding”type=”tns :Hotel”>
<soap :bindingstyle=”document”
transport=http ://schemas.xmlsoap.org/soap/http/ >
<operationname=”process”>
<soap :operation
soapAction=http ://hotel.localhost/process/ >
<input>
<soap :bodyuse=”literal”/ >
< /input>
<output>
<soap :bodyuse=”literal”/ >
< /output>
< /operation>
< /binding>

<servicename=”HotelService”>
<portname=”HotelPort”binding=”tns :HotelBinding”>
<soap :addresslocation=http ://localhost :8082/ode/processes/Hotel/ >
< /port>
< /service>
< /definitions>
```

E.3 Service-Location Voiture

```
<?xmlversion="1.0"? >
<definitionsname="Loc-Voiture"
targetNamespace=http ://loc-voiture.localhost
xmlns :tns=http ://loc-voiture.localhost
xmlns :plnk=http ://docs.oasis-open.org/wsbpel/2.0/plnktype
xmlns=http ://schemas.xmlsoap.org/wsdl/
xmlns :soap="http ://schemas.xmlsoap.org/wsdl/soap/" >

<types>
<schemaattributeFormDefault="unqualified"elementFormDefault="qualified"
targetNamespace=http ://loc-voiture.localhost
xmlns=http ://www.w3.org/2001/XMLSchema>

<elementname="Loc-VoitureRequest">
<complexType>
<sequence>
<elementname="Id-Loc"type="string"/ >
<elementname="Date-Prise"type="date"></ element>
<elementname="Date-Remise"type="date"></ element>
<elementname="Prix-Loc"type="double"></ element>
< /sequence>
< /complexType>
< /element>
<elementname="Loc-VoitureResponse">
<complexType>
<sequence>
<elementname="Date-Depart"type="date"/ >
<elementname="Date-Retour"type="date"></ element>
<elementname="Lieu-Loc"type="string"></ element>
< /sequence>
< /complexType>
< /element>
< /schema>
< /types>
<messagename="Loc-VoitureRequestMessage">
<partname="payload"element="tns :Loc-VoitureRequest"/ >
< /message>
<messagename="Loc-VoitureResponseMessage">
<partname="payload"element="tns :Loc-VoitureResponse"/ >
< /message>
```

```
<portTypename="Loc-Voiture">
<operationname="process">
<inputmessage="tns :Loc-VoitureRequestMessage"/>
<outputmessage="tns :Loc-VoitureResponseMessage"/>
</operation>
</portType>
<plnk :partnerLinkTypename="Loc-Voiture">
<plnk :rolename="Loc-VoitureProvider"portType="tns :Loc-Voiture"/>
</plnk :partnerLinkType>

<bindingname="Loc-VoitureBinding"type="tns :Loc-Voiture">
<soap :bindingstyle="document"
transport="http ://schemas.xmlsoap.org/soap/http"/>
<operationname="process">
<soap :operation
soapAction="http ://loc-voiture.localhost/process"/>
<input>
<soap :bodyuse="literal"/>
</input>
<output>
<soap :bodyuse="literal"/>
</output>
</operation>
</binding>

<servicename="Loc-VoitureService">
<portname="Loc-VoiturePort"binding="tns :Loc-VoitureBinding">
<soap :addresslocation="http ://localhost :8080/Loc-Voiture"/>
</port>
</service>
</definitions>
```

E.4 Service-Bank

```
<?xmlversion="1.0"? >
<definitionsname="Bank"
targetNamespace=http ://bank.localhost
xmlns :tns=http ://bank.localhost
xmlns :plnk=http ://docs.oasis-open.org/wsbpel/2.0/plnktype
xmlns=http ://schemas.xmlsoap.org/wsdl/
xmlns :soap="http ://schemas.xmlsoap.org/wsdl/soap/" >

<types>
<schemaattributeFormDefault="unqualified"elementFormDefault="qualified"
targetNamespace=http ://bank.localhost
xmlns=http ://www.w3.org/2001/XMLSchema>

<elementname="BankRequest">
<complexType>
<sequence>
<elementname="Prix-Total"type="double" / >
< /sequence>
< /complexType>
< /element>
<elementname="BankResponse">
<complexType>
<sequence>
<elementname="Liste-Prix"type="double" / >
< /sequence>
< /complexType>
< /element>
< /schema>
< /types>
<messagename="BankRequestMessage">
<partname="payload"element="tns :BankRequest" / >
< /message>
<messagename="BankResponseMessage">
<partname="payload"element="tns :BankResponse" / >
< /message>
<portTypename="Bank">
<operationname="process">
<inputmessage="tns :BankRequestMessage" / >
<outputmessage="tns :BankResponseMessage" / >
< /operation>
```

```
< /portType>

<plnk :partnerLinkTypename="Bank">
<plnk :rolename="BankProvider"portType="tns :Bank" / >
< /plnk :partnerLinkType>

<bindingname="BankBinding"type="tns :Bank">
<soap :bindingstyle="document"
transport=http ://schemas.xmlsoap.org/soap/http/ >
<operationname="process">
<soap :operation
soapAction=http ://bank.localhost/process/ >
<input>
<soap :bodyuse="literal" / >
< /input>
<output>
<soap :bodyuse="literal" / >
< /output>
< /operation>
< /binding>

<servicename="BankService">
<portname="BankPort"binding="tns :BankBinding">
<soap :addresslocation=http ://localhost :8082/ode/processes/Bank/ >
< /port>
< /service>
< /definitions>
```

E.5 Composition-des-servicesReservation

```
<?xmlversion="1.0"encoding="UTF-8"standalone="no"? >
<definitionsxmlns="http ://schemas.xmlsoap.org/wsdl/"xmlns :plnk=
"http ://docs.oasis-open.org/wsbpel/2.0/plnktype"xmlns :soap="http ://sche-
mas.xmlsoap.org/wsdl/soap/"xmlns :tns=
"http ://reservation"xmlns :vprop="http ://docs.oasis-
open.org/wsbpel/2.0/varprop"xmlns :wsdl= "http ://bank.localhost"xmlns :wsdl1=
"http ://vol.localhost"xmlns :wsdl2="http ://loc-
voiture.localhost"xmlns :wsdl3=
"http ://hotel.localhost"name="Reservation"targetNamespace=http ://reservat
ion>
<plnk :partnerLinkTypename="bankPLT">
<plnk :rolename="bankRole"portType="wsdl :Bank"/ >
< /plnk :partnerLinkType>
<plnk :partnerLinkTypename="volPLT">
<plnk :rolename="volRole"portType="wsdl1 :Vol"/ >
< /plnk :partnerLinkType>
<plnk :partnerLinkTypename="locationPLT">
<plnk :rolename="locationRole"portType="wsdl2 :Loc-Voiture"/ >
< /plnk :partnerLinkType>
<plnk :partnerLinkTypename="hotelPLT">
<plnk :rolename="hotelRole"portType="wsdl3 :Hotel"/ >
< /plnk :partnerLinkType>
<importlocation="BankArtifacts.wsdl"namespace=http ://bank.localhost/ >
<importlocation="VolArtifacts.wsdl"namespace=http ://vol.localhost/ >
<importlocation="Loc-VoitureArtifacts.wsdl"namespace=http ://loc-
voiture.localhost/ >
<importlocation="HotelArtifacts.wsdl"namespace=http ://hotel.localhost/ >
<types>
<schemaxmlns="http ://www.w3.org/2001/XMLSchema"attributeFormDefault=
"unqualified"elementFormDefault="qualified"targetNamespace=http ://reser-
vat
ion> <elementname="ReservationRequest">
<complexType>
<sequence>
<elementname="Date-D"type="date"/ >
<elementname="Nom-H"type="string"/ >
<elementname="Prix-Total-B"type="string"/ >
<elementname="Lieu-visite"type="string"/ >
< /sequence>
< /complexType>
< /element>
<elementname="ReservationResponse">
```

```
<complexType>
<sequence>
<elementname="Billet-Vol"type="string"/>
<elementname="Contrat-H"type="string"/>
<elementname="Contrat-V"type="string"/>
<elementname="Prix-t"type="string"/>
</sequence>
</complexType>
</element>
</schema>
</types> <messagename="ReservationRequestMessage">
<partelement="tns :ReservationRequest"name="payload"/>
</message>
<messagename="ReservationResponseMessage">
<partelement="tns :ReservationResponse"name="payload"/>
</message>

<portTypename="Reservation">
<operationname="process">
<inputmessage="tns :ReservationRequestMessage"/>
<outputmessage="tns :ReservationResponseMessage"/>
</operation>
</portType>
<plnk :partnerLinkTypename="Reservation">
<plnk :rolename="ReservationProvider"portType="tns :Reservation"/>
¡/plnk :partnerLinkType>

<bindingname="ReservationBinding"type="tns :Reservation">
<soap :bindingstyle="document"transport="http ://sche-
mas.xmlsoap.org/soap/
http"/>
<operationname="process">
<soap :operationsoapAction="http ://reservation/process"/>
<input>
<soap :bodyuse="literal"/>
</input>
<output>
<soap :bodyuse="literal"/>
</output>
</operation>
</binding>

<servicename="ReservationService">
<portbinding="tns :ReservationBinding"name="ReservationPort">
<soap :addresslocation="http ://localhost :8082/ode/processes/Reservation"/>
</port>
</service>
</definitions>
```