الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي و البحث العلمي

جامعة وهران للعلوم و التكنولوجيا محمد بوضياف

THÈSE

En vue de l'obtention du Diplôme de Doctorat

Présentée par : Ali KADDOURI

Intitulé

Cloud Computing : Gestion autonome et garantie du niveau de service de bout en bout

Faculté	***: Géni Electrique***
Département	***: Electronique***
Domaine	***: Science et Technique***
Filière	***: Électronique***
Intitulé de la Formation	***: Technologie moderne de l'infirmation et de communication***

Devant le Jury Composé de :

Membres de Jury	Grade	Qualité	Domiciliation
ABDELAZZIZ. OUAMRI	Professeur	Président	USTO
MUSTAPHA. GUEZOURI	Professeur	Encadreur	Université Oran 1
MOKHTAR. KECHE	Professeur	Co-Encadreur	USTO
MOHAMED. OUSLIM	Professeur	Examinateur	USTO
GHALEM. BELALEM	Professeur	Examinateur	Université Oran 1
KADDOUR EL BOUDADI LAHOUARI	MCA	Examinateur	USTO
-	-		-

Année Universitaire : 2016/2017

A mes chers parents

Remerciements

Avant tous, je remercie le bon DIEU de m'avoir donné le courage et la volonté pour accomplir ce travail de recherche. Je tiens à exprimer ma profonde reconnaissance au directeur de thèse Mr GUEZOURI Mustapha, Professeur à l'université d'Oran 1 Ahmed BENBELLA pour son suivi, sa patience, sa disponibilité, ses nombreux conseils et ses critiques constructives pour l'élaboration de ce travail.

Je tiens à remercier tous ceux qui ont contribué de près ou de loin à l'accomplissement de ce travail en particulier Mr Nader MBAREK professeur à l'université de bourgogne pour son accueil et ses orientations durant mes premières années; Mr M. KECHE et aussi Mr A. OUAMRI pour m'avoir permis d'être membre de leur laboratoire LSI. Je remercie ainsi l'ensemble des enseignants et doctorants de laboratoire LSI pour leur patience et leur esprit de partage et plus en particulier MmeImen TALEB.

De plus, je remercie mes parents qui sont les premiers qui m'ont donné support durant tous mon cursus de doctorat. Mes vifs remerciementsà ma famille, mes amis et mon entourage.

Je tiens à remercier les membres du jury qui ont accepté de juger ce travail. Je cite Mr le professeur A. OUAMRI (USTO), Mr le professeur M. KECHE, Mr le professeur M. OUSLIM (USTO). Mr Le professeur G. BELALEM (université Oran 1) et finalement Mr le docteurL. KADDOUR EL BOUDADI. Je remercie tous ceux qui ont accepté de faire partie du jury et de consacrer leurs temps précieux afin d'évaluer ce travail.

خلاصة

أظهرت عدة منظمات أكاديمية وصناعية اهتماما بتكنلوجيا السحابة الحاسوبية. ان النموذج الاقتصادي للسحابة الحاسوبية كان من أكثر الخصائص المرغوبة. وجدت المنظمات ان السحابة الحاسوبية هو أقل تكلفة مقارنة مع أي نموذج آخر للأنظمة الموزعة التي استعملت سابقا. يعطي نموذج السحابة الحاسوبية وهما للمنظمات وكأن الموارد بالداخل هي غير منتهية. عكس ذلك، هذا بطبيعة الحال غير صحيح وعلى مزودي خدمات الاعلام الآلي في الغيوم أن يملأ كل تزايد في طلبات الموارد في أي وقت لأي مستعمل. هذه الفلسفة تحرر المنظمات من أي استثمار وصيانة لأنظمة تكنولوجية المعلوماتية. من الجانب الآخر، مزودو خدمات الاعلام الآلي في الغيوم سوف يتلقون كل عمليات الصيانة المكثفة ومسؤولية توفير الخدمة لكل الخدمات من أجل أن تضمن بنسبة جد عالية على عدم كسر او خرق عقد مستوى الخدمة. ان مزودي خدمات السحابة الحاسوبية يعلمون عن مثل هذه المشاكل المتعلقة بخرق عقود مستوى الخدمات. من أجل أكثر جودة ودقة وكذلك الابتعاد عن مشكل عدم توفر الخدمة الكافية لتسيير والرد على كل الطلبات المتزايدة حول الموارد. فتح مجتمع السحابة الحاسوبية محاور جديدة للبحث التي ترغب في تبادل الموارد بين مزودي خدمات السحابة الحاسوبية. الفكرة في حد ذاتها واضحة لكن تحتاج عمل كثير للوصول الى مبتغى والأهداف. التحدي الذي يقابل التواصل والتشغيل البيني يكمل في ان النموذج الجديد لأنظمة الموزعة يفقر معيار موحد. أغلب الأبحاث التي قيمت فيما يتعلق بهذا الموضوع اين عرضت عدة هندسيات التي يمكن اعتبارها كمعيار موحد لكل مزودين خدمات السحابة الحاسوبية لكي يتبعوه. هناك فئة قليلة من الذين تكلموا في أبحاثهم عن طرق التواصل. كباحثين، لقد رغبنا في اللجوء الى الطريقة الثانية لحل جزء من المشكل. لقد قررنا تصميم وانجاز بروتوكول مواصلات الذي بإمكانه معالجة مفاوضات الموارد مع ضمان نسبة من المستوى للخدمة. البروتوكول طور في محيط شبيه بالحقيقة. أكثر نقطة اهتماما التي أخذناها بعين الاعتبار في التصميم هو قدرت البروتوكول بمعالجة اي خدمة خاصة متعلقة بميدان معين. كمثال الى جانب ما كنا مهتمين به والذي كان ميدان المهمات الفضائية وامكانية استعماله فوق بنية تحتية متركزة على نموذج السحابة الحاسوبية. لقد طبقنا البروتوكول المصمم على برنامج يخص المهمات الفضائية. النتائج كانت حسنة وأظهرت رغبة كبيرة في طريقة تمكن البروتوكول من تكامل مع خدمات خاصة بميادين معنية وكذلك معدلات نسبية للمعلومات المتبادلة بين المتحدثين. في المستقبل سنأخذ بعين الاعتبار استراتيجية لإضافة جانب الأمن للبروتوكول

Abstract

Cloud Computing gains interests between academic and industrial organisations. The economic model of Cloud Computing was the most feature of interest. Organisations find the Cloud Computing less costly than any other model used before. Cloud Commuting gives to these organisations an illusion that resource inside is unlimited. However, this is not true and Cloud Provider must fulfil any increase in resource demand at any time for any user. This philosophy liberate organisation on any IT investment and maintenance. In the other side, Cloud Computing providers will receive all the load of maintenance and service availability on them. To guarantee the SLA, Cloud provider knew this problem of SLA violation. For more reliability and to bypass the problem of inability to manage and fulfil all increase in resource demand, the Cloud community opened a new axis of research which interest in inter-Cloud resource exchange. The idea is clear but needs a lot to do to reach the final objective. The challenges for the inter-cloud communication and interoperability is the fact that this new model of distributed systems lack standardization. Most of researches done regarding this topic were kind of proposed architectures that may be a standard for every Cloud provider to follow. Rarely are those who speak about communication process. As researchers, we were interested on the second way to resolve part of the problem. We decide to design and implement a communication protocol, which handles resources negotiation with a guaranty of certain level of service. The protocol was developed in an environment similar to the reality. The most important point we take on the design is the ability of the protocol to handle any domain specific services. As an example to what we have interest on was space mission design above a Cloud Computing infrastructure. We applied the protocol on a space mission application. Results were good and show important interest on how the protocol is able to handle multiple specific domain service as well as the data overload generation. We take consideration to develop a strategy to add security aspect to the protocol.

Résumé

Le Cloud Computing gagne d'intérêt entre les organisations académiques et industrielles. Le modèle économique du Cloud Computing était le caractère le plus intéressant. Les organisations trouvent que le Cloud Computing est beaucoup moins chère que n'importe quel modèle précédemment utilisé. Le Cloud Computing donnait á ces organisations une illusion que les ressources sont infinies. Par contre, ceci n'est pas vrai. Les fournisseurs de service Cloud doivent répondre à l'augmentation des demandes des ressources pour n'importe quel utilisateur et á n'importe quel moment. Cette philosophie libère les organisations de l'investissement en IT et á sa maintenance. Dans l'autre sens, les fournisseurs de service Cloud vont subir tous la charge de maintenance et la responsabilité de disponibilité de service pour garantir une haute probabilité de garantie de SLA. Les fournisseurs Cloud savaient ce problème de violation de SLA. Pour plus de fiabilité et pour palier le problème d'incapacité de manager et garantir les augmentations des demandes des ressources, la communauté Cloud ouvre un nouvel axe de recherche qui s'intéresse à l'échange des ressources entre fournisseurs Clouds. Cette idée est claire mais nécessite beaucoup de travail á faire. Pour atteindre les objectifs finaux. Le challenge pour la communication et l'interopérabilité inter-Cloud est en fait ce nouveau modèle de système distribué qui manque de standardisation. La plus part des recherches faites concernant cette topic étaient genre d'architectures proposées qui peuvent être un standard pour chaque fournisseur Cloud á suivre. Rares sont ceux qui parlaient sur le processus de communication. Comme chercheurs, nous nous sommes intéressés à la deuxième façon pour résoudre une partie du problème. On a décidé de concevoir et d'implémenter un protocole de communication qui traite la négociation des ressources avec une garantie de niveau de service. Le protocole était développé dans un environnement similaire á la réalité. Le point le plus important pris en considération lors de la conception est la capacité de protocole de traiter n'importe quel service spécifique á un domaine particulier. Comme exemple, nous nous sommes intéressés à la conception des missions spatiales basées sur une infrastructure Cloud Computing. Nous avons appliqué le protocole sur une application spatiale. Les résultats étaient bons et montraient un intérêt important sur la façon du protocole de s'adapter aux services spécifiques á des domaines particuliers, de même pour les résultats en termes de taux de génération des données. On prendra en considération dans des travaux futurs le développement d'une stratégie pour apporter l'aspect sécurité au protocole.

Introduction générale

L'apparition d'internet a fait un chemin à de nouveaux modèles des systèmes distribués. Ces modèles ont permis la collaboration de plusieurs entités géographiquement distribuées. Avant l'apparition de Cloud Computing, il existait ce qu'on appelle les Grids Computing. Le Grid Computing est une collection des machines hétérogènes géographiquement distribuées. Chaque machine exécute une tache particulière. L'ensemble des taches s'exécutant dans différentes machines vont aboutir à un seul objectif pour lequel la Grid a été créée. Le plus souvent, un intergiciel est associé à une Grid afin de permettre la communication, le provisionnement et le contrôle des ressources. Comme les Grids Computing, le Cloud Computing est un nouveau modèle de système distribué basé sur plusieurs technologies déjà existantes tel que le web service, architecture orienté service (Service Oriented Architecture, SOA), le Grid Computing. De plus, la virtualisation était un facteur majeur dans l'apparition de Cloud.

Le Cloud Computingou l'informatique dans le nuage se repose sur un modèle économique suivant la règle "payement à l'usage". Un fournisseur Cloud met à la disponibilité de ses clients un ensemble de ressources logique (logiciels, machines virtuelles) et physique (serveurs de stockage, ressources réseau). Un utilisateur Cloud peut accéder au service quand il veut, où il veut, avec la performance qu'il veut. Tout ça, sans se mêler de la configuration, la gestion et la maintenance de ces ressources. Un point important, c'est que le Cloud donne une illusion que les ressources sont infinies, chose qui n'est pas vrais. Le Cloud se présente sous diffèrents types de déploiement. On trouve le Cloud public, le Cloud privé, le Cloud hybride ou mixte et le Cloud communautaire. De plus, les services sont classés dans trois catégories dont le Software as a Service (SaaS), Plateforme as a Service (PaaS) et Infrastructure as a Service (IaaS). Dans le marché de Cloud, plusieurs grandes entreprises très connues dans le domaine informatique se sont intéressées à cette technologie dont on cite Microsoft avec ses services comme Microsoft Azure et Google avec le service Google App Engine. Les avantages de Cloud se résument dans sa flexibilité et scalabilité ainsi son modèle économique qui offre un cout startup trop faible pour toute nouvelle entité qui souhaite intégrer une infrastructure IT.

Le domaine Cloud ouvre de nouveaux axes de recherche. Les clients et les fournisseurs Cloud se mettent d'accord sur ce qu'on appelle un agrément de niveau de service (Level Service Agreement, SLA). Le client et le fournisseur ne doivent pas violer cet agrément. Les données et les services résident dans le Cloud en dehors de l'infrastructure de client, alors le fournisseur doit assurer la disponibilité des données et le temps de réponse. Le fournisseur fait recours aux techniques de réplication. La réplication engendre le problème de non cohérence des données. Avec le Cloud, les services peuvent être couplés pour atteindre un objectif. Ces services sont accessibles par plusieurs utilisateurs à la fois alors qu'ils partagent une même ressource. Ceci engendre le problème d'ordonnancement. En outre, le Cloud cause le problème de sécurité et confidentialité des données des clients stockées chez le fournisseur. Avec le Cloud, on économise beaucoup en termes de hardware mais de l'autre coté la bande passante peut devenir trop couteuse si la quantité des données échangées est trop grande. Le

Cloud manque de standard, les fournisseurs Cloud définissent leurs propres technologies à l'intérieur de leur infrastructure. Ceci freine la collaboration entre les fournisseurs Cloud et interdira au client de migrer d'un fournisseur à un autre. De plus, pour permettre la collaboration entre Clouds, des protocoles et des interfaces communes sont nécessaires. L'interopérabilité est un challenge qui vient en deuxième position après celui de la sécurité dans le Cloud.

Le concept de Cloud permet d'avoir des ressources illimitées, dans le cas où le fournisseur atteint ses limites, il sera plus capable d'assurer au client autant de ressource, ce qui fait que le fournisseur va violer le SLA. Ceci était toujours un problème pour les fournisseurs. La bonne façon de résoudre ces problèmes est de collaborer avec d'autres fournisseurs Cloud offrant le même service. La collaboration inter-cloud nécessite un moyen de communication qui étend la capacité de garantir le niveau de service à travers plusieurs Clouds. Dans ce cadre, notre travail se présente dans la forme d'un protocole de communication entre Clouds permettant la découverte des services, l'échange d'information de routage, propagation des notifications, réservation de ressource et demande de service. Notre protocole définit un ensemble de messages de contrôle. Le message UPDATE permet l'échange d'information concernant les services et les chemins menant à ces services par tous les fournisseurs Clouds. Les informations reçus sont stockées dans une table qui sera une entrée pour le processus de sélection. Le message REQUEST (requête) est utilisé par un fournisseur Cloud demandant à un autre fournisseur de lui mettre à sa disposition un service bien déterminé. La demande peut contenir certains critères. Une demande de service peut traverser plusieurs fournisseurs intermédiaires avant d'atteindre sa destination finale. Les Clouds intervenants peuvent signaler une erreur. La signalisation des erreurs se fait avec un message ERROR. Une session de demande de service est considérée comme complète qu'après l'échange des quatre messages (REQUEST, ACCEPTANCE, REQUEST CONFIRMATION ; RESERVATION CONFIRMATION). L'ensemble des messages sont encapsulés dans un entête équivalent à celui de BGP [6]. Le protocole définit aussi une table de routage très simple et extensible. La table contient l'identifiant de Cloud ; Identifiant de service ; lien (les identifiants des Clouds intermédiaires) et interface de sortie. À la base le meilleur lien est celui avec le chemin le plus court. Les résultats en termes de temps de convergence et quantité des données générées étaient bonnes et paraissent si petite comparée aux ressources énormes que le Cloud détient. Pour montrer aussi la portabilité et la possibilité d'utiliser le protocole dans différents domaines, nous nous sommes intéressés à une application dans le domaine spatiale. L'application montre un exemple sur la capacité de protocole d'adapter des besoins spécifiques à un domaine.

La thèse est organisée comme suite : Chapitre 1 ; on donne une définition de Cloud ; l'historique de Cloud. On explique les caractéristiques de Cloud. On passe à la description de différentes architectures de Cloud et les différents types des services. On donne par la suite ; une vue globale sur les challenges de Cloud et on tire de ceci notre problème visé. Chapitre 2 ; on présente un état de l'art sur les travaux existants et qui ont un lien avec le problème étudié. Nous aborderons plus particulièrement la gestion de la qualité de service de bout en

bout dans internet et le Cloud. Le chapitre 3 détaille notre contribution. On décrit l'ensemble des structures de protocole et ses opérations. Le chapitre 4 présente les résultats ; l'interopération et la discussion. Chapitre 5 présente les résultats de l'utilisation de protocole dans une application spatiale en décrivant aussi les besoins du domaine spatiale et comment le Cloud et le protocole répond à ces besoins.

nous donnerons à la fin une conclusion sur les travaux et les résultats obtenus et nos perspectives.

Chapitre 1

Le Cloud Computing

1.1Introduction

Le Cloud est une nouvelle technologie qui porte plusieurs avantages garce au modèle économique basé sur la facturation à l'usage et sa flexibilité. Nous avons consacré ce chapitre pour le définir ; présenter l'historique de Cloud, parler sur ses caractéristiques et les avantages ; détailler les différentes architectures de Cloud et aussi les types des services (classification).Toute nouvelle technologie est supposée ouvrir des axes de recherche pour résoudre certains problèmes et challenges ; alors le Cloud en fait partie et pour cela, nous allons décrire les différents problèmes qui font l'objet des recherches actuelles. Ensuite s'intéresse au problème qui vise à proposer une solution pour la garantie de niveau de service de bout en bout.

1.2Définition de Cloud:

Pour le Cloud Computing, il existe plusieurs définitions données. (A.Roussignol, 2014) a publié 25 différentes définitions de Cloud Computing [1]. Les experts plus souvent dans leurs articles font référence à celle NIST (National Institut of Standards and Technology) [2] qui définisse le Cloud comme suit : Le Cloud Computing est un modèle pour permettre un accès convenable ; omniprésent, à la demande au réseau partageant un ensemble de ressource de calcule configurable (ex. Réseaux, Serveurs, Stockages, Applications et Services) qui peuvent être rapidement contrôlées et libérées avec une gestion facile.

(R. Buyya, 2008) ; a définit le Cloud comme suit : c'est un type de système parallèle et distribué composé d'une collection d'ordinateurs virtualisés et interconnectés qui sont provisionnés dynamiquement et présentés comme une ou plusieurs ressources unifiés basés sur le SLA établie entre le fournisseur et le client [3].

(P. McFedries, 2008) défini le Cloud comme suite : non seulement nos données mais aussi nos applications et logiciels résident dans le Cloud et qui sont accessibles avec différents terminaux (Ordinateur, PDA, Smartphone …etc.). La puissance de calcul est offerte grâce à la virtualisation. Cette puissance est alimentée de grands Datacenters [2].

A partir des connaissances et les différentes définitions existantes, nous avons donné une définition au Cloud comme suit: c'est une nouvelle technologie de système distribué. Des ressources physiques et logiques sont mises à la disponibilité des clients accessibles à distance (via internet). La facturation d'utilisation des ressources est basée suivant la règle « paiement à la demande ». La maintenance de la globalité du système est à la charge du fournisseur. La technologie de virtualisation est à la base d'apparition de ce modèle.

1.3Contexte Historique :

Le Cloud Computing est une évolution qu'une révolution. C'est en 2006 que le Cloud faisait apparition lorsqu'Amazon dévoila sa version d'essai d'Elastic Cloud Computing (EC2). L'offre était destinée aux développeurs qui ne souhaitaient pas disposer de leur propre infrastructure informatique et qui la louaient donc à Amazon, en passant par Internet [3].

L'expression est devenue populaire qu'on 2007. Avec les années ; les grandes sociétés d'IT se sont lancées dans le marché de Cloud dont Google avec Google App Engine [4] et Microsoft avec Azure [5].

Le Cloud Computing est le résultat de convergence de plusieurs technologies existantes dont le Grid Computing, Utility Computing, l'informatique à la demande, Architecture Orientée Service et en particulier la virtualisation.

1.4Les Caractéristiques de Cloud:

La force de Cloud vient de ses caractéristiques qui englobent les avantages des technologies précédentes. Les experts donnent plusieurs caractéristiques. On se contente seulement des cinq caractéristiques les plus communes.

1.4.1 Accès à la demande, self-service Access:

- *Accès à la demande* : les utilisateurs peuvent accéder quand ils veulent, où ils veulent. Les utilisateurs peuvent allouer et libérer des ressources suivant les besoins.

- *Self-service* : les utilisateurs peuvent accéder au Cloud, allouer des ressources, configurer, Controller et déployer des services comme serveur de temps, serveur de base de données sans une intervention humaine du coté de fournisseur.

1.4.2 Accès à un réseau important :

Les fournisseurs Cloud déploient ses infrastructures devant le backbone du réseau internet. Ceci permettra de diminuer le temps de réponse à une valeur trop petite. De plus, une grande capacité de bande passante est utilisée par le Cloud pour permettre une satisfaction au plus grand nombre d'utilisateurs. Les ressources sont accessibles à travers des mécanismes standards pour promouvoir l'utilisation par des plateformes clientes lourdes ou légères (Téléphone portable, Tablet, Ordinateur portable ou Ordinateur de bureau).

1.4.3 Réservoir des ressources (resource pooling) :

Les ressources des fournisseurs de Cloud sont rassemblées pour servir plusieurs utilisateurs en utilisant un modèle multi-tenants avec différents ressources physiques ou logiques (virtuelles) dynamiquement affectées et de-affectées selon le besoin et la demande du client. Généralement, le client n'a pas de contrôle ou connaissance sur l'endroit exact des ressources fournies. Le client par contre peut spécifier l'endroit à un haut niveau d'abstraction (ex. Pays, état...etc.). Un exemple de ressource est le stockage, calcul intensif, mémoire et bande passante.

1.4.4 Elasticité rapide :

Les ressources peuvent être louées et libérées dynamiquement. Le Cloud répond très rapidement au besoin de client (application). Le client peut stocker sans limite. Le cloud résout le problème de pointe et l'idle state (ressource disponible mais pas utilisée).

Un serveur de vente peut facilement provisionner autant de ressource de calcul pour faire face au pointe qui apparaissent soudainement dans des moments connus ou inconnus (Noel). Une foi fini, le client relâche ces ressources. Les ressources au regard de client sont infinies.

1.4.5 Service mesuré (payement à l'usage) :

L'utilisation des ressources est rapportée pour donner une transparence sur la facturation. Les ressources sont associées avec des métriques. Un client paye uniquement les ressources qu'il utilise. Pour une messagerie électronique, on peut facturer selon le nombre de compte créé par le client.

D'autres caractéristiques implicites comme le model multi-tenant, efficacité de prix …etc peuvent être ajoutées au caractéristiques de Cloud.

1.5 Modèle de déploiement de Cloud

Le modèle de Cloud peut être implémenté à l'intérieur d'une infrastructure existante. De ce fait l'infrastructure va bénéficier des avantages de cloud. Il existe quatre modèles de déploiement de Cloud.

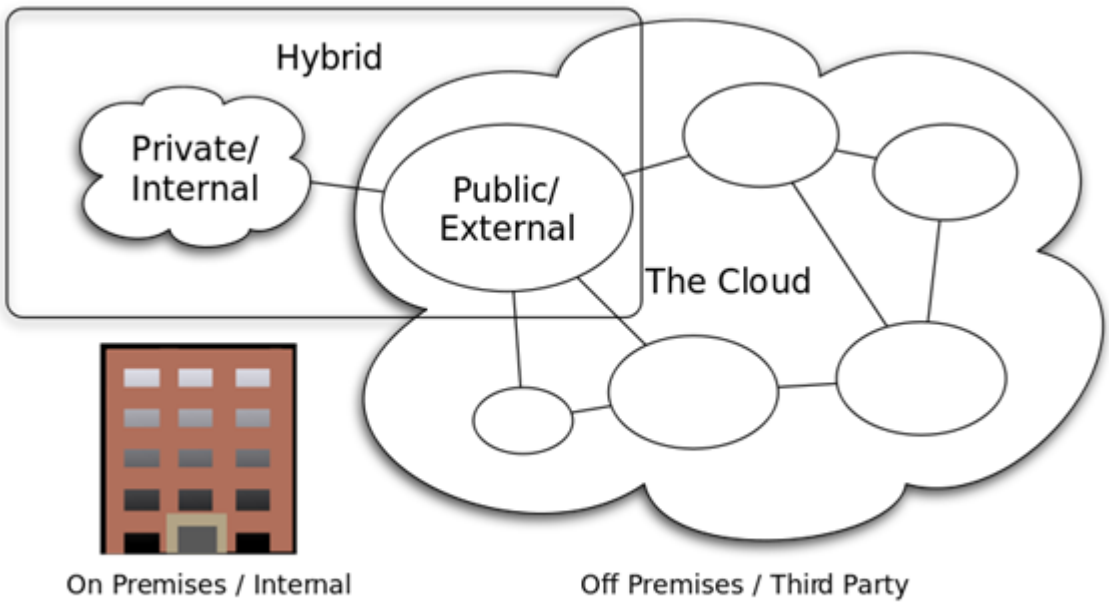

Figure 1: Modèle de déploiement de Cloud privé, publique et hybride ;https://en.wikipedia.org/wiki/Cloud_computing#/media/File:Cloud_computing_types.svg

1.5.1 Cloud privé :

Dans ce modèle, l'infrastructure Cloud a la propriété d'une seule entité (Entreprise/Organisation). L'infrastructure est soit à l'intérieur du siège de l'organisation ou bien dans un lieu distant. De plus, elle peut être gérée par les employées de l'organisation ou bien part des professionnelles/sous-traitants spécialistes. L'infrastructure est seulement utilisée par les membres de l'organisation. Un Cloud Computing est similaire à un réseau

local (LAN) sauf que le Cloud privé installe les technologies Cloud à l'intérieur comme la virtualisation et les Datacenters.

Le Cloud privé convient les organisations qui peuvent trouver l'aspect sécurité très important peuvent car la confidentialité des données à l'intérieur de Cloud est plus assurée que dans un Cloud public.

Les avantages de cloud privé par rapport aux caractéristiques de Cloud sont les suivants :

- ***Haute sécurité et confidentialité :*** le Cloud privé peut garantir la sécurité de ses données et applications en utilisant des techniques comme la restriction d'Accès à certain connexions venant de l'extérieur de firewall. Les liens et connexions privés permettront de garder les opérations discrètes.

- ***Plus de contrôle :*** comme le Cloud est accessible seulement par une seule organisation. Cette dernière peut configurer et gérer en parallèle l'infrastructure avec les besoins pour achever une solution réseau sur mesure.

1.5.2 Cloud public :

Dans ce modèle, une organisation investit dans une grande infrastructure offrant plusieurs services, déployés dans un ou plusieurs endroits. Les services sont offerts au grand public ou à des grandes sociétés industrielles. Ce type de cloud est le plus courant. L'accès aux ressources se fait via un réseau (souvent internet). Les avantages du Cloud publique sont les suivants :

-***Scalabilité :*** le Cloud publique dispose d'un grand réservoir de ressources. Les applications qui s'exécutent dedans peuvent répondre d'une façon transparente aux fluctuations de l'activité.

- ***Facturation :*** les services du Cloud publique emploie le modèle de facturation (payement à la consommation) par lequel le client sera capable d'accéder aux ressources selon son choix et il paie seulement ce qu'il a consommé. Donc, on évite la perte de capacité.

- ***Fiabilité:*** le grand nombre des serveurs et réseaux impliqués dans la création d'un Cloud public ainsi que la redondance permettront à un service de Cloud public de continuer à s'exécuter même si un composant tombe en panne. D'une autre façon, il y'a pas de problème d'un seul point de défaillance qui mettrait le service d'un cloud public vulnérable

- ***Independence de lieu*** : la disponibilité des services de Cloud publique à travers internet assure que les services sont disponibles partout où se trouve le client.

1.5.3 Cloud mixte :

Ce modèle fusionne le Cloud privé et le Cloud public. Une organisation possède sa propre infrastructure Cloud alors que cette dernière est connectée à un Cloud public qui utilise les mêmes technologies pour permettre la portabilité et l'interopérabilité. Les avantages de Cloud mixte sont comme suit :

- ***Scalabilité :*** tandis que le Cloud privé offre un certain niveau de scalabilité, les services de Cloud publique vont offrir une scalabilité peux limitée car les ressources sont

tirées à partir d'un grand réservoir. En migrant les opérations et les données non confidentielles au Cloud public, ceci permettra à l'organisation de bénéficier de la scalabilité de Cloud diminuant les demandes sur le Cloud privé.

- ***Efficacité de cout :***comme le Cloud public offre un cout trop bas. De ce fait un Cloud privé peut bénéficier de ceci toute en gardant les données et opérations sensibles sécurisées à l'intérieur du Cloud privé.

- ***Sécurité :***La partie privé d'un Cloud mixte (hybride) peut offrir la sécurité pour les opérations et les données lorsque elle est nécessaire, mais aussi peut régulièrement satisfaire les besoins de traitement et stockage.

1.5.4 Cloud communautaire :

C'est une infrastructure Cloud mise à la disponibilité d'une communauté professionnelle bien déterminée qui peut inclure des partenaires, des scientifiques travaillant sur un objectif commun.

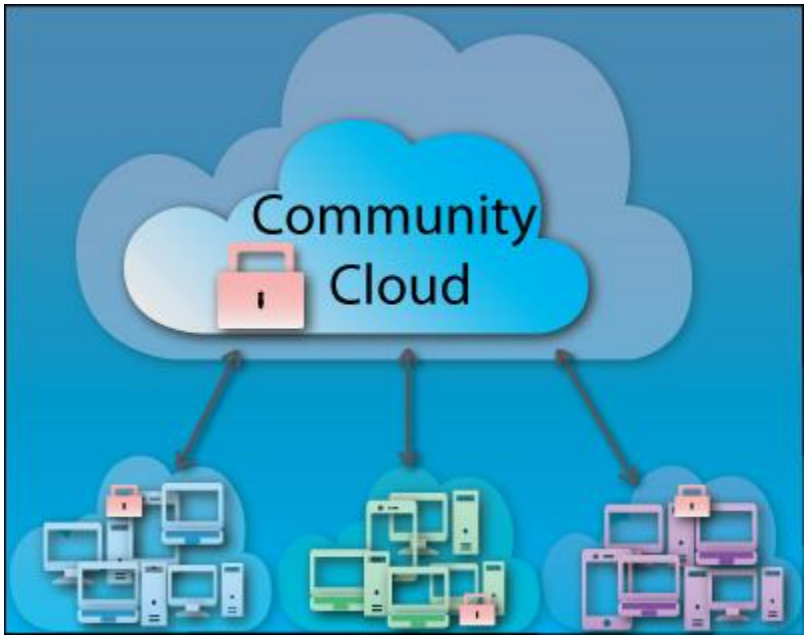

Figure 2 : Cloud Communautaire :
http://nauges.typepad.com/.a/6a00d8345167aa69e201b8d13c2b36970c-pi

1.6 Les Types des Services Cloud :

Les différents services sont classés en trois principales catégories. Les différents catégories se superposent l'une sur l'autre donnant une pile de couche. Sa présentation ressemble à celui de la pile protocolaire TCPIP. Il existe trois couches principales, chacune inclue un type de service.

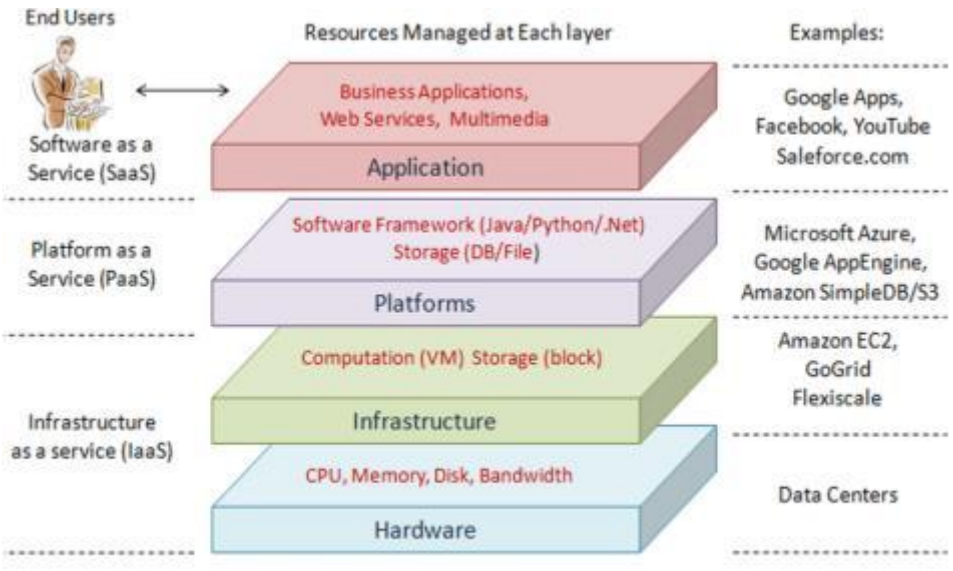

Figure 3 Les Couches des différents types des services de Cloud ;https://amagsmb.wordpress.com/2014/06/23/cloud-computing-part-3-architecture/

1.6.1 Infrastructure as a Service :

C'est la couche qui fait référence au provisionnement à la demande des ressources de l'infrastructure. Souvent, c'est en termes de machines virtuelles. Cette couche est devisée en deux sous-couches hardware et virtuelle :

- **Couche hardware** : responsable de la gestion des ressources physiques de Cloud comme les serveurs, routeurs, commutateurs, énergie et système de refroidissement

- **Couche virtuelle** : .sert à créer un réservoir de ressources de stockage et de calcul en utilisant différentes techniques et outils de virtualisation tel que Xen[7], KVM[8] et VMware[9]. La virtualisation des ressources est la technologie majeure qui permettra de présenter les ressources à la demande.

Les services de la couche infrastructure peuvent être utilisés par des entreprises clientes pour créer une solution fiable et efficace en diminuant le cout. La complexité et les dépenses sur la gestion du hardware sont externalisées vers le fournisseur de Cloud. Le client peut ajouter, configurer et retirer des ressources si nécessaire plus facilement. La couche infrastructure peut être utilisée comme :

- **structure d'entreprise :** les entreprises peuvent installer une infrastructure à l'intérieur de leur siège qui va former un Cloud privée. Les ressources sont exploitées suivant le modèle Cloud (à la demande). Les données et les opérations sont faites à l'intérieur de l'infrastructure. Ceci évitera le problème de sécurité et confidentialité.

- **Hébergement :** Des sites web peuvent être hébergés dans des machines virtuelles et que ces dernières peuvent changer de configuration pour répondre aux besoins en termes de stockage et traitement des requêtes. L'hébergement dans le Cloud peut bénéficier des techniques de réplication qui assura la disponibilité de service et des données.
- **Réseaux virtuels :** avec l'avancement dans la technologie de virtualisation, c'est devenu facile de créer des réseaux virtuels, on peut interconnecter plusieurs machines virtuelles pour créer la topologie souhaitée qui répondra aux besoins. Une telle configuration peut être offerte à des chercheurs dans le domaine des réseaux et systèmes répartis qui ont besoin de plusieurs entités réseaux pour tester leurs solutions. OpenGeni [10] est un exemple de telles expériences.

Les utilisateurs des services de la couche infrastructure peuvent bénéficier de la scalabilité, baisse de cout, une facturation à l'usage, un accès omniprésent, une sécurité physique des serveurs et disponibilité de service grâce à la réplication et redondance.

1.6.2 Platform as a service :

Cette couche fournit des applications plateforme comme les systèmes d'exploitation, les langages et les environnements de programmation. On trouve comme exemple Google App Engine, Microsoft Windows Azure et saleforce.com.

Cette couche vient au-dessus de la couche infrastructure as a service. Le but de cette couche est de minimiser la charge de déploiement des applications directement dans le conteneur de machines virtuelles. A titre d'exemple, Google propose des API multiples liées aux opérations de stockage, base de données, commerce et les applications web.

Parmi les avantages d'utilisation des outils de la couche plateforme as a service est la facilite de travailler dans une équipe où les membres peuvent être dans différents lieux. Il permet aussi aux utilisateurs non-experts de programmer des applications beaucoup plus facile.

1.6.3 Software as service :

La couche la plus haute dans la hiérarchie est la couche Software as a Service. Cette couche contient différentes applications spécifiques à un domaine. Elle peut être une application de messagerie, éditeur de texte, de comptabilité ou tout autre type. L'avantage est que les applications bénéficient de caractéristique d'élasticité des ressources d'une façon automatique pour achever une bonne performance, avec un cout faible. Les applications et logiciels de Cloud sont modulaires ce qui permettra aux utilisateurs de les intégrér dans des nouvelles applications.

La hiérarchie des types de service Cloud est similaire au modèle OSI ou bien TCP/IP. Les logiciels dans la couche software as a Service est loués au client qui veut les utiliser. Les utilisateurs inscrivent sur le logiciel au lieu de l'acheter. Les avantages des SaaS sont l'absence d'investissement au lancement de projet, Mise à jour automatique, compatibilité avec plusieurs terminaux (téléphone, ordinateur …etc.).

En général, les couches inferieures de Cloud offrent un service aux couches supérieures ; d'une autre façon, les couches supérieures sont considérées comme des clients des couches inférieures. L'utilisateur final est le client de la couche SaaS dans laquelle, il trouvera les logiciels nécessaires pour réserver toute ressource dont il a besoin.

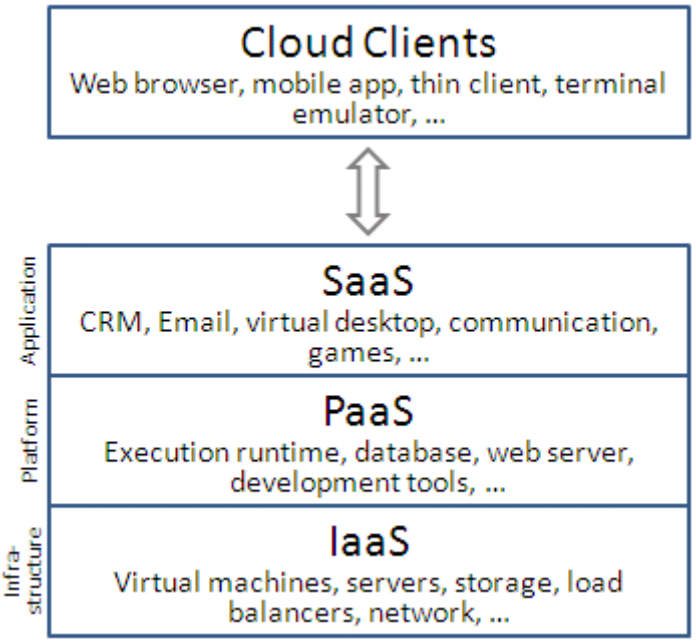

Figure 4: Accessibilité aux couches de Cloud ; http://www.mazikglobal.com/blog/cloud-computing-stack-saas-paas-iaas/

1.7 Les problèmes et challenges de Cloud

Le Cloud a connu un grand succès grâce à son modèle économique qui a aidé les utilisateurs à minimiser les couts et l'investissement liés aux services IT.

Le Cloud est le résultat de rassemblement de plusieurs technologies existantes dont le GridComputing, utility Computing, autonomicComputing et la virtualisation. Ces technologies étaient des champs d'expérience des chercheurs. Plusieurs problèmes étaient étudies. Le Cloud a fait en sorte qu'il a adopté ces axes de recherche et il est devenu un domaine qui souffre d'un ensemble de problèmes qui font l'objet des travaux de recherche actuel liés au Cloud.

1.7.1 Migration des machines virtuelles :

La puissance de Cloud revient à l'utilisation avancé de la technologie de virtualisation. Grace à cette technologie, il est devenu possible de créer une machine virtuelle avec une configuration sure mesure. Les machines virtuelles ont la possibilité de migrer d'une ressource à une autre. Les machines virtuelles peuvent être distribuées sur plusieurs ressources physiques ou consolider sur une seule machine pour balancer les charges.

La migration des machines virtuelle fait appelle au concept d'ordonnancement. L'ordonnancement a toujours causé des difficultés pour les processus dans les systèmes d'exploitation ou dans les GridComputing. Cette difficulté existe aussi dans le Cloud dont l'ordonnancement concerne les machines virtuelles. Les algorithmes d'ordonnancement doivent optimiser l'utilisation des ressources physiques et augmenter la performance des applications s'exécutants dans le Cloud.

La migration de la mémoire passe par trois phases principales :

- **pré-phase :** durant l'exécution de la machine virtuelle dans la ressource initiale, des pages mémoire sont envoyés à la nouvelle source destinatrice. Pour la persistance du transfert, les pages mémoire modifiées durant la migration est renvoyée.

- **arrêter et copier phase :** la machine virtuelle source est arrêtée, les pages sont copiées dans la nouvelle machine virtuelle résidant dans la ressource destinataire.

- **post-phase** : dans cette étape la machine est prête à s'exécuter. Dans le cas où une page est manquante, la nouvelle machine est considérée comme erronée par rapport à la machine virtuelle initiale.

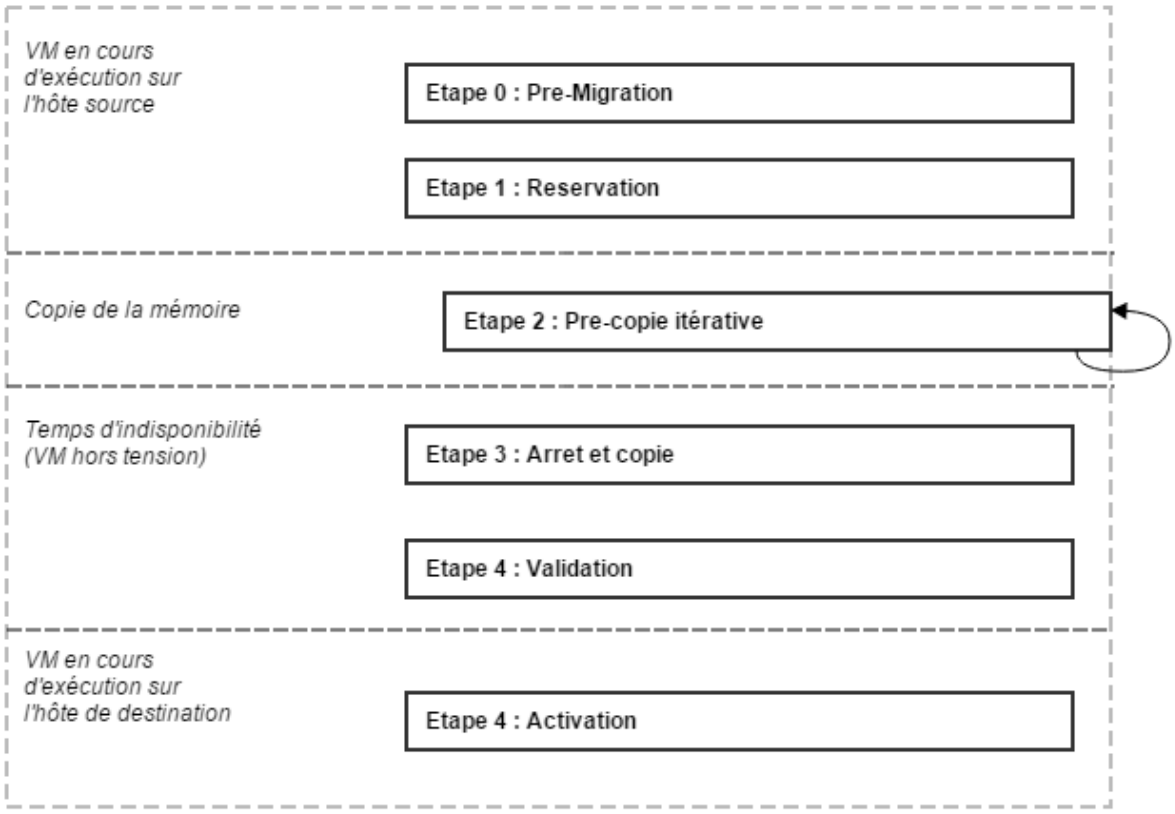

Figure 5: Les phases principales de migration de mémoire ; https://fr.wikipedia.org/wiki/Migration_de_machines_virtuelles

Les machines virtuelles sont attachées à la mémoire, le réseau et le disque de stockage. La grande difficulté lors de la migration de la machine virtuelle en temps réel c'est le mapping

d'une machine à l'autre. La migration à froid engendre moins de problèmes que la migration à chaud. Le seul problème c'est que les services qui s'exécutent dans cette machine virtuelle ne seront pas disponibles durant le temps de migration. La migration à chaud vient pour résoudre le problème de la migration à froid.

La migration des machines virtuelles doit prendre en considération l'estimation des performances, perte des performances durant la migration et le taux de consommation d'énergie. De plus l'aspect de sécurité est toujours présent. Durant la migration de la machine il peut y avoir un risque sur la confidentialité.

1.7.2 Consolidation des serveurs :

La migration à chaud des machines virtuelles a permis aux administrateurs réseaux de définir des stratégies d'assemblement au maximum les machines virtuelle dans le minimum de ressource phasique. La consolidation des serveurs est un problème de logique combinatoire. La consolidation des serveurs est une approche pour maximiser l'utilisation, tandis que l'énergie est minimisée dans un environnement. Avec les machines virtuelles, la charge peut être distribuée de façon àceque les applications et machines virtuelles peuvent être migrées à partir de plusieurs ressources physiques sous-utilisées, exécutant peut d'application à une seule ressource physique. De cette façon plusieurs ressources physiques n'auront aucune machine virtuelle qui s'exécute dedans seront libéré. Les ressources physiques libérées peut être misent en mode idle (économiser l'énergie) (P.D Patet, 2014) donna un survey sur la consolidation des serveurs dans le Cloud Computing [11].

En logique combinatoire, le problème d'optimisation de consolidation des serveurs est vu comme celui de problème de bin-packing. Le problème de pin-packing est un problème NP-complet [12].

Des méthodes heuristiques sont proposées pour résoudre ce genre de problème [13] [14] [15]. L'optimisation de consolidation des serveurs doivent prendre en considération plusieurs aspects. La consolidation des serveurs ne doit pas causer une dégradation des performances des applications. Cette dégradation peut être liées au partage de plusieurs machines virtuelles et applications media de transfert qui causera une congestion et donc une dégradation. La consolidation des serveurs est une tache autonome, elle doit donc répondre automatiquement aux fluctuations qui peuvent arrivées subitement dans l'environnement.

SPECvirt_sc2010 est un banc de teste standard pour mesurer la capacité des solutions de virtualisation pour la consolidation des serveurs [16]. Elle exécute un ensemble de machines virtuelles composées de serveur web, serveur de messagerie et serveur Java pour compléter des transactions. Des métriques QoS sont utilisées pour estimer les performances comme le temps de réponse.

1.7.3 Sécurité et confidentialité :

Par rapport à la couche Software as a Service, la sécurité est liée beaucoup au fournisseur Cloud, à cause du degré d'abstraction. La couche Software as a service est située dans un niveau d'abstraction très élevée avec un contrôle minime. D'autre part, la couche Plateforme

as a Service offre une grande extension et contrôle au client à cause de l'abstraction basse dans le système. Comme chaque couche dépend de l'autre, n'importe quelle attaque sur une couche aura un impact sur la couche supérieur. Cette dépendance entre les couches est une source de vulnérabilité. Un fournisseur Software as a Service peut louer un environnement de développement depuis un fournisseur Plateforme as a Service, qui lui-même alloue des ressources depuis un fournisseur Infrastructure as a Service. Chaque fournisseur est responsable de sécuriser ses services. Cette philosophie engendre une combinaison inconsistante en termes de sécurité. Si une attaque arrive, ça devient une ambiguïté dans la détermination de lieu de la faille.

1.7.3.1 Sécurité dans la couche Software as a Service :

Les services de la couche Software as a Service comme la messagerie, logiciel de conférence, ERP, CRM et autres sont accessibles typiquement via internet en utilisant un exploiteur web. Le flux dans les applications web peut créer des vulnérabilités pour les applications Software as a Service. Le web était toujours compromis avec des programmes qui peuvent voler des données trop sensibles. De ce fait, les problèmes de sécurité dans la couche software as a Service ne se différencie pas de celui des applications web existantes sauf que les techniques traditionnelles ne protègent pas efficacement le système contre les nouvelles failles. Des nouvelles techniques sont nécessaires.

Le projet Open Web Application Security Project (OWASP) a identifié dix menaces les plus critiques dans la sécurité des applications web [17].

-***Multi-tenants*** *:* dans ce type de consommation de ressource, une seule instance servira tous les utilisateurs. Cette approche permet d'utiliser les ressources efficacement. De plus, les données sont stockées dans une même base de données. Ce qui fait, que le risque de fuite des données entre les tenants est trop élevé. Alors, des politiques de sécurité sont nécessaires pour assurer que les données sont gardées séparées d'un utilisateur à l'autre [18].

- ***Sécurité des données*** : la sécurité des données est un problème commun pour toutes les technologies, mais avec le Software as a Service ceci est devenu un problème majeur. Les utilisateurs comptent sur leurs fournisseurs pour la sécurité de leurs données. La sauvegarde des données est un point critique pour pouvoir récupérer les données en cas de catastrophe. Parfois, les fournisseurs adhèrent à d'autres fournisseurs offrant tel service ce qui engendre une multiplication de risque et non possibilité d'assurer la confidentialité.

- ***Accessibilité*** : l'Access aux applications Software as a Service via internet en utilisant plusieurs types de terminaux (ordinateur, téléphone …etc.) exposent le client à beaucoups plus de menaces. the Cloud Security Alliance a publié un document sur le Cloud mobile et les menaces liées aux vols des données due au non sécurité des medias de communication, failles dans les systèmes d'exploitations et les applications officielles et autres types de failles et attaques [19].

1.7.3.2 Sécurité dans la couche Platform as a Service :

La sécurité de Platform as a Service dépend principalement d'un réseau et un explorateur sécurisé. Le fournisseur Platform as a Service est responsable de sécuriser la pile des applications de cette couche qui contient le noyau d'exécution des applications des clients.-

- ***Multipartenaires :*** les services des fournisseurs peuvent utiliser des services des autres fournisseurs. De ce fait, le Platform as a Service hérite tous les failles de sécurité liées aux données et réseaux [20]. La sécurité dépendra alors, de client, de son fournisseur et la troisième partis. Ceci élargira le contexte de sécurité et le rend encore vulnérable.

- ***Cycle de vie de développement :***les développeurs des applications Cloud doivent prendre en considération les mises à jour régulières des logiciels de plateforme. La conception des applications doivent être flexible pour permettre un changement rapide. Le changement dans la plateforme peut compromettre la sécurité des applications. Les données sont stockées dans différents endroits et peuvent suivre plusieurs politiques de sécurité et confidentialité. Alors, les développeurs doivent être au courant de tous ces aspects.

La couche Platform as a Service nécessite des recherches considérables pour aboutir à un ensemble de techniques adaptées à la flexibilité de Cloud et ses caractéristiques pour augmenter la capacité des procédures et protocoles de protections et sécurisations.

1.7.3.3 Sécurité de la couche Infrastructure as a Service :

La couche Infrastructure as a Service fournit des ressources physiques tel que des serveurs, des réseaux, stockage, et autres sous forme de systèmes virtuels. Ces ressources sont accessibles via internet. Les utilisateurs ont plus de contrôle sur l'aspect sécurité comparé aux autres couches (PaaS, SaaS). Les ressources virtuelles sont contrôlé par un hyperviseur seulement le fournisseur Cloud a l'autorité sur cet outils. Les fournisseurs de Infrastructure as a Service doivent assurer un bon contrôle. Les menaces peuvent subvenir dans les opérations de création, modification et mobilité des ressources virtuelles [21].

- ***virtualisation*** : la virtualisation permet de créer, copier, partager, migrer et libérer des machines virtuelles. Ceci introduit des nouvelles opportunités et failles au pirate. La sécurité des machines virtuelles deviennent aussi importante que les machines physiques. Les machines virtuelles sont vulnérables à tous les types des attaques pour des structures classiques [22]. La sécurité des machines virtuelles se repose sur elle-même et la machine physique dont elle l'héberge.

- ***contrôle des machines virtuelles*** : les gestionnaires des machines virtuelles ou hyperviseurs sont responsables de l'évolution des machines virtuelles. Dans le cas où cet outil est compromis, il causera la compromission des machines virtuelles.

De plus, la virtualisation peut migrer des machines virtuelles d'un serveur physique à un autre. Un intrus peut utiliser cette caractéristique pour migrer la machine vers un serveur physique non malicieux.

-***Ressources partagées :***les machines virtuelles sur le même serveur peuvent partagées, l'unité de calcul (CPU), la mémoire, E/S (entrées/sorties) et autres. Les partages

des ressources entre les machines virtuelles causent des vulnérabilités. Une machine virtuelle peut avoir accès à une autre sans avoir besoin de compromettre le hyperviseur.

- ***Réseaux virtuelles*** : les composants réseaux sont partagés par plusieurs tenants. La plus part des hyperviseurs utilisent des liens virtuelles pour que les machines virtuelles puissent communiquer directement afin d'augmenter la performance et la fiabilité de communications. Les techniques de sniffing et spoofing peuvent être utilisées dans le cas où le gestionnaire des machines virtuelles utilise des routeurs et des commutateurs dans la communication entre les machines virtuelles.

1.7.4 Portabilité et interopérabilité :

L'interopérabilité peut être définie comme la mesure de degré dont plusieurs systèmes ou composants peuvent travailler ensemble correctement. Institute of Electrical and Electronics Engineers (IEEE) définit l'interopérabilité comme la capacité d'un système ou un produit de travailler avec autres systèmes facilement et sans faire un effort du coté de client. L'interopérabilité devient possible avec l'implémentation des standards [23].

Dans le contexte de cloud, l'interopérabilité peut être vue comme la capacité que plusieurs types de fournisseurs Cloud hétérogènes de s'identifier en terme des applications, les interfaces des services, la configuration, l'authentification, autorisation, formatage des données …etc. L'idéal pour l'interopérabilité dans le Cloud est de standardiser les interfaces à des niveaux d'abstraction différentes.

La portabilité est la capacité de déplacer une entité à partir d'un système à un autre et la faire fonctionner sans aucun effort. La portabilité est divisée en deux catégories :

- ***portabilité des données :*** c'est la capacité de transférer facilement des données d'un fournisseur Cloud à un autre. Le service source devra transférer les données avec le formatage que le Cloud destinataire l'accepte. Même si le format ne consiste pas, la transformation entre les deux formats des deux Clouds communicants doit être simple à achever grâce à la mise en disponibilité des outils nécessaires pour ce processus.

La portabilité des données permet à un utilisateur Cloud de transférer les données d'un Cloud à un autre. Ceci se fait avec des interfaces communes ou standards (FTP). La syntaxe et la sémantique sont en fait aussi très importantes dans la portabilité des données. C'est possible que la translation se fasse avec des outils en cas de différence dans le formatage et la sémantique définie dans les deux Clouds.

- ***portabilité des applications :*** la portabilité ne s'arrête pas juste au déplacement des données. Plutôt, de même pour les applications qui se définissent comme étant la capacité de faciliter le transfert des applications d'un Cloud à un autre et l'exécuter correctement. Il est acceptable d'adapter légèrement l'application en recompilant par exemple de code source de l'application. Mais il n'est pas acceptable de faire un important changement dans le code de l'application pour qu'elle puisse s'exécuter dans la nouvelle plateforme.

Le problème principal qui bloque l'interopérabilité est le non standardisation des interfaces et les APIs des services Cloud. Chaque Cloud utilise des APIs beaucoup plus spécifique pour son infrastructure.

L'interopérabilité diminue à chaque fois qu'on va d'une couche inferieure à une couche supérieure. Pour la couche Infrastructure as a Service, on trouve le Cloud Data Management Interface standard (CDMI). Un standard, qui spécifie les protocoles d'auto-provisionnement, administration et l'accès aux services de stockage [24]. Le problème d'interopérabilité persiste dans la couche supérieur Software as a Service. Un déplacement d'une application Service as a Service d'un cloud à un autre avec les mêmes fonctionnalités va nécessiter un changement dans les interfaces. Une approche pour lever ce problème est d'utiliser une troisième partie « Cloud Broker ». Ce dernier est responsable de transcrire les interfaces d'un Cloud à plusieurs autres interfaces des autres Clouds. Comme les applications sont exécutées sur une plateforme, il devient nécessaire et difficile au même temps de concevoir une application pour un Cloud et l'exécuter dans un autre Cloud sans avoir une intervention complexe. En pensant à l'interopérabilité, il est important de prendre en considération l'aspect sécurité des données et des applications. Il faut donc apporter une amélioration dans un sens de trouver un compromis qui satisfait l'ensemble des critères. Grace à l'établissement d'un mécanisme fiable d'interopérabilité, les clients ne feront plus face au problème de lock-in. Les clients seront capables de basculer d'un service à un autre suivant les besoins. Le client peut même utiliser plusieurs services de différents fournisseurs Clouds à la fois et combiner les services des fournisseurs Clouds pour des raisons de coût, confidentialité, performance, ou disponibilité. D'autres parts, les fournisseurs cloud peuvent collaborer en temps réels pour augmenter les performances et éviter de violer le SLA avec les clients finaux.

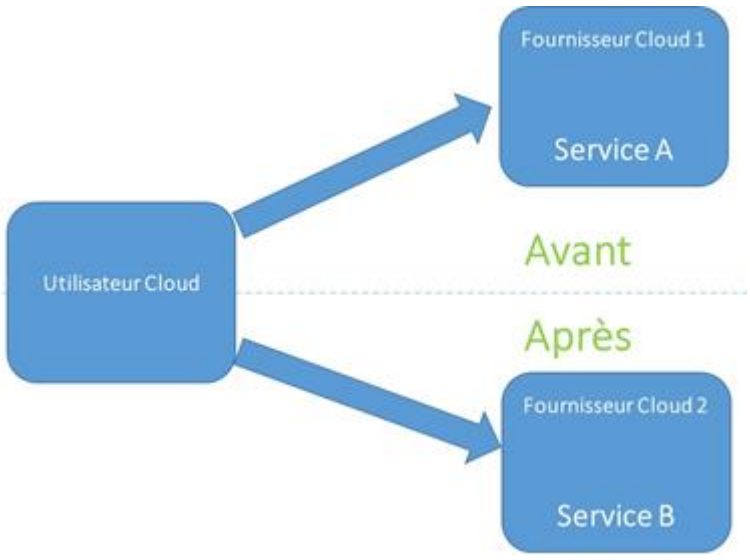

Figure 6 : Interopérabilité - Basculement de client entre fournisseurs Clouds

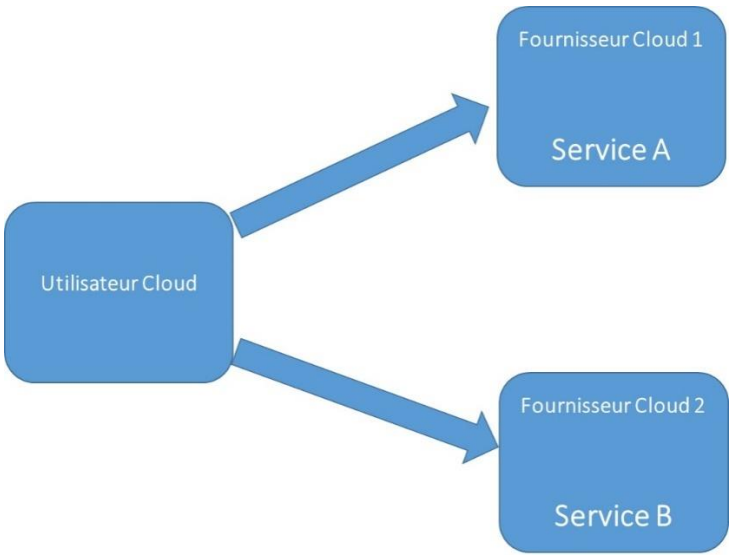

Figure 7 : Interopérabilité - Utilisation des services de plusieurs fournisseurs Clouds en même temps

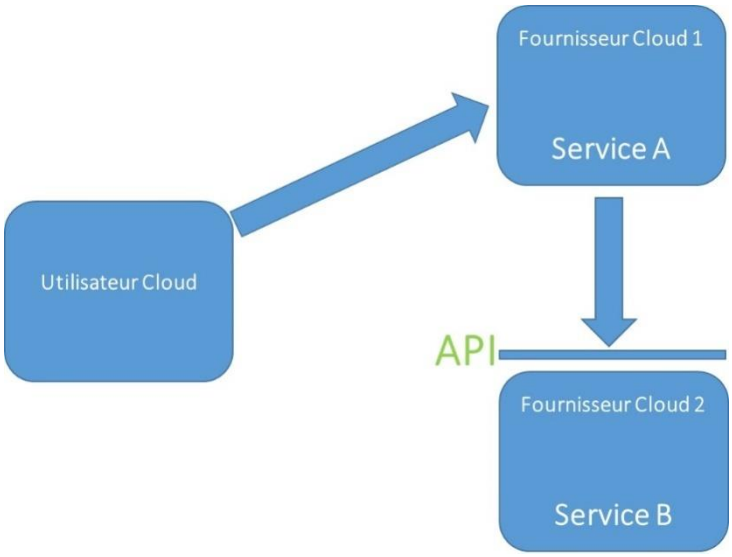

Figure 8 : Interopérabilité - Coopération entre fournisseurs Clouds

1.8 La communication inter-Cloud :

Parmi les différents axes de recherche dans le domaine de Cloud Computing, la sécurité reste la plus grande contrainte qui freine plusieurs compagnies de rejoindre ce nouveau paradigme. Les clients n'osent pas transférer leurs données dans un Cloud publique auquel aucune technologie, protocole ou mécanisme propre pour garantir que leurs données soient protégées de toute insécurité, d'accès non autorisé, fuite, non disponibilité, dépendance et autres problèmes.

L'interopérabilité vient en deuxième lieu après celle de la sécurité. Sans l'interopérabilité, les clients de Cloud seront victime de dépendance à un seul fournisseur Cloud. Le client fait épreuve la première fois de passer d'une infrastructure classique interne par rapport au client vers un nouveau modèle qui est le Cloud. Cette tache nécessite du temps, un cout et une mise

en disponibilité d'une cellule rien que pour planifier les taches et les plans de l'action de passage.

Les expectations des clients ne sont pas toujours les mêmes avec la réalité. Parfois, le passage vers une structure Cloud public devient indésirable principalement. Dans tel cas un utilisateur choisira de changer de fournisseur, alors que cette tâche ne peut être accomplit car la portabilité dans le Cloud ne connait pas des standards ou des interfaces communes pour tel objectifs. Des initiatives sont lancées pour créer un standard ouvert comme celui d'Open Cloud Interoperability Framework [25].

L'interopérabilité a intéressé plusieurs chercheurs. Chacun travaille sur un point essentiel qui mènera à permettre aux fournisseurs Clouds de collaborer entre eux d'une façon autonome. Comme décrit précédemment, plusieurs scenarios sont réalisables dans le cas où l'interopérabilité et la portabilité devient réaliste. Parmi ces scenarios, il y a le cas de collaboration de plusieurs fournisseurs Cloud de point- à-point (peer-to-peer) pour avoir un réseau multi-Cloud, multi-lien et multi-choix.

La garantie du niveau de service d'un client à l'intérieur d'un Cloud nécessite des précautions à prendre du coté de fournisseur. En revanche, cette garantie est contrôlable car les ressources physiques et logiques sont à la propriété de fournisseur qui peut intervenir sur les mécanismes, la configuration, prise de décision pour éviter toute violation de SLA avec leurs clients.

La garantie du niveau de service à travers plusieurs Cloud sous l'autorité de différents administrateurs et propriétaires devient une tâche complexe et délicate. Cette problématique est classée parmi les dérivations de l'axe de recherche lie à l'interopérabilité.

Vu que les Clouds ne parlent pas les mêmes langages, et qu'ils n'utilisent pas les mêmes technologies, il devient important de penser à une solution qui permet de rendre la communication entre Clouds possible et avec laquelle les fournisseurs peuvent automatiquement quand ils veulent exploiter d'autres services existants chez un autre fournisseur Cloud. Nous nous sommes intéressés plus précisément à cette partie qui concerne le grand problème de communication entre Clouds pour s'entre-aider où un fournisseur peut demander, réserver, exploiter, choisir et libérer des ressources existantes dans des Clouds voisins où distants d'une façon autonome.

Principalement, la communication nécessite un protocole bien défini qui permet aux fournisseurs Clouds de l'utiliser pour la signalisation et l'application des actions nécessaires pour garantir le niveau de service de point-à-point dans l'environnement Cloud, implanter et exécuter un protocole de communication inter-cloud pour l'objectif d'automatiser les opérations et garantir aux maximum le SLA.

1.9 Feuille de route :

Notre travail vise à résoudre le problème de limitation des ressources au sein d'un fournisseur de service Cloud. Les fournisseurs sont obligés de respecter leurs contrats avec leurs clients. Ces clients exigent souvent qu'ils doivent disposer des ressources nécessaires quel que soit sa taille et à n'importe quel moment. .nous proposons une solution (un produit : protocole), ce protocole doivent être exécuté dans les routeurs des fournisseurs de service Cloud. Notre

solution sert à permettre la communication entre les fournisseurs Cloud, échanger les informations des services, ressources, qualité et négociation. Nous utilisons le langage de programmation C. nous exécuterons le programme dans des machines virtuelles générées par l'émulateur NETKIT. On se lance notre recherche depuis des protocoles existants comme BGP, MPLS, RSVP et des travaux proposés par d'autres chercheurs qui ont eux aussi conçu des protocoles de QoS de boot en boot. Nous ferons référence aux solutions qui existent dans les autres paradigmes comme le réseau internet et les Grid computing. Nous estimons les performances de protocole avec des expériences répétitives. Nous estimons le temps la quantité des données.

1.10 Conclusion :

Le Cloud représente un virage important que l'informatique aille au-dessus. Le passage au Cloud va mettre ses clients à découvrir une nouvelle vue sur les concepts de programmation, conception et utilisation des outils informatique. Le Cloud est une nouvelle technologie qui cache des contraintes dont elles composent les grandes lignes de la recherche. Un important élément ; la standardisation des interfaces et l'autorisation aux différents fournisseurs de pouvoir communiquer. Les travaux qui concernent la garantie du niveau de service de bout en bout existaient avant dans d'autres paradigme (internet, GridComputing). Ces recherches feront l'objet de chapitre suivant.

Chapitre 2

La garantie de niveau de service de bout en bout

2.1 Introduction :

Malgré le buzz que la technologie Cloud fait, elle reste toujours une évolution des technologies précédentes. Que ce soit le Cloud, Grid, Cluster ou autres types de systèmes distribués. La base de communication entre les entités est le modèle TCP/IP.

TCP/IP est un modèle conceptuel standard qui permet la communication entre plusieurs machines réseaux. Grace à ses caractéristiques de flexibilité et scalabilité. Il est devenu le principe basique de communication dans le grand réseau internet. En observant le réseau internet à un niveau trop bas, il apparait comme étant plusieurs réseaux privés interconnectés entre eux via des routeurs. Les communications IP utilisent la fragmentation pour subdiviser les données de grande taille à des paquets de petite taille. Les données se transfèrent de routeur en routeur jusqu'elles atteignent la destination finale. D'où le transfert des données de bout en bout dans le réseau internet.

Le réseau internet était le premier paradigme où des recherches ont été menées pour apporter la qualité de service à travers plusieurs chemins.Temps de réponse, bande passante, équilibrage entre interfaces et autres étaient les principes de QoS dans le réseau internet. Comme le Cloud utilise le réseau internet (public) ou bien toujours le modèle conceptuel TCP/IP (privé), alors ce dernier s'engage aussi aux problèmes de QoS dans les réseaux.

Le Cloud est le successeur de Grid où les machines physiques collaborent pour augmenter les puissances de calcul et de stockage. Les inergiciels ont la responsabilité de permettre aux intervenants de communiquer via des mécanismes fiables augmentant la performance.

Dans les trois points précédents cités on remarque qu'il y a une relation entre les travaux de recherches qui se concentrent sur la garantie de niveau de service ou la qualité de service dans le Cloud. Nous présentons dans ce chapitre l'ensemble des travaux qui explorent ce sujet dans les trois concepts internet, Grid et Cloud Computing.

2.2 La Qualité de service dans le réseau internet:

Le réseau internet fournit seulement un service *best effort* [26] [27]. Le trafic est routé vers sa destination. Le best effort n'offre pas une garantie d'envoi de trafic au destinataire. Le réseau internet est devenu une infrastructure commerciale. Les utilisateurs et entreprise payent des services plus élaborés et fiables pour rendre leurs trafics prioritaires et rapidement traités, on parle donc de la qualité de service qui doit être implémentés dans ce réseau et plus précisément dans les routeurs et commutateurs.

Les besoins des applications distribuées se diffèrent. Il y'a des applications qui nécessitent une grande bande passante pour transférer une grande quantité de donné comme les serveurs Web. Il y'en a d'autres qui ont besoin d'un délai d'attente minimum

pour assurer une demande …etc. Le flux de données des applications peut donc être groupé en plusieurs classes.

Des mécanismes sont nécessaires pour fournir la qualité de service dans le réseau internet. Actuellement tous les routeurs et commutateurs de différentes marques sont dotés d'un ensemble de mécanismes de la gestion de la qualité de service. Plusieurs solutions sont fournies dont en trouve :

- differentiated service

- Traffic Engineering

- Constraint based Routing

- Multi-Protocol Label Switching (MPLS)

- Integrated service – Resource Reservation Protocol (RSVP)

- Autres techniques.

Ces concepts ont été une base d'exploitation. Pour cela, nous essayerons d'expliquer chaque concept et plus en détails le protocole RSVP et MPLS.

2.2.1 Services différenciés (Differentiated service):

Le service différencié est une autre approche qui apporte la qualité de service dans le réseau internet. Il utilise le champ TOS inclus dans l'entête IP [28], comme illustré dans la figure 9. Le champ TOS est lui-même composé de trois sous-champs. Les trois premiers bits appelés précédence, les trois bits suivants sont appelé respectivement D, T et R pour indiquer des priorités en qualité de délais, débit et fiabilité (perte des paquets). Ces trois bits sont nommées « champs de service différencié ». Les deux bits restants sont réservé pour une utilisation ultérieure. La figure 10 montre le contenu du champ TOS du header IP.

L'utilisation des six bits (précédence et DS) permettra d'augmenter le nombre de classe aux 64 possibilités.

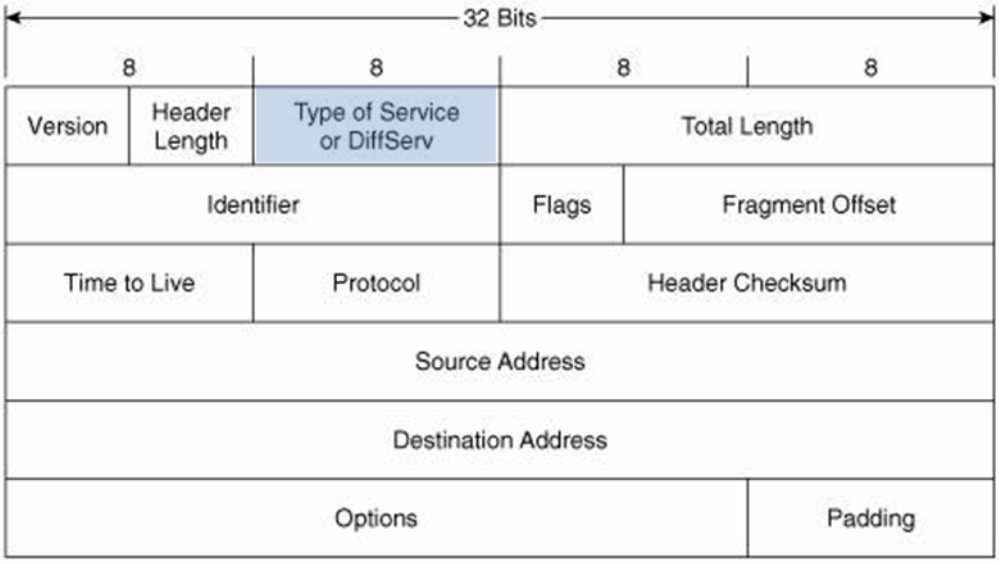

Figure 9 : Le champ Type De Service -TOS- dans le header IPv4 ; https://advancedinternettechnologies.wordpress.com/ipv4-header/

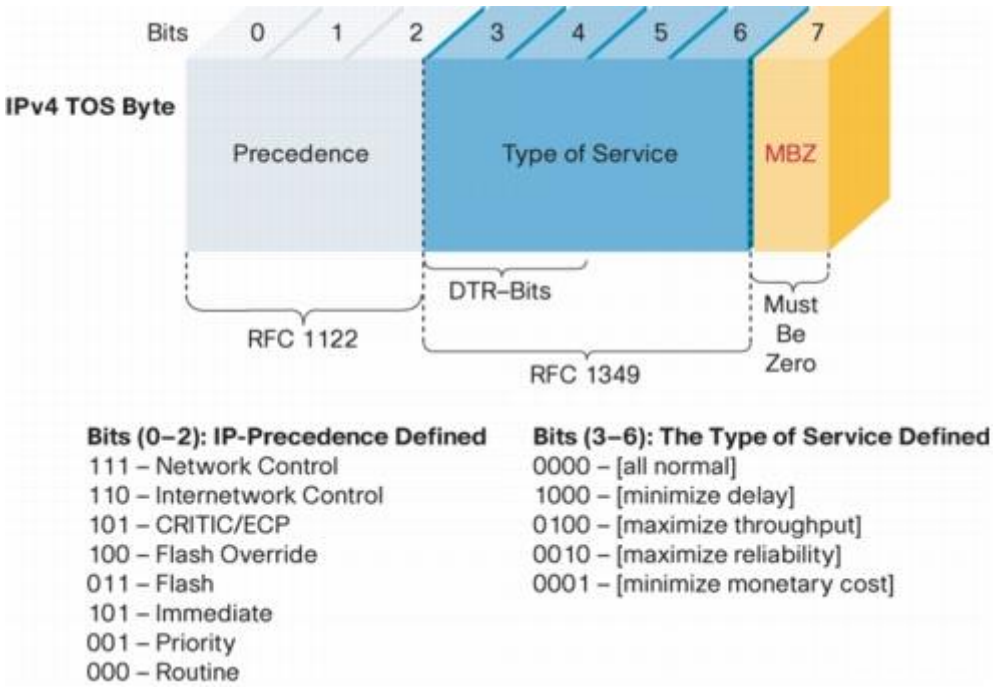

Figure 10: Structure de Champs Type De Service -ToS- ; http://www.cisco.com/en/US/technologies/tk543/tk766/technologies_white_paper09186a00800a3e2f.html

Pour recevoir une certaine qualité de service, le client doit indiquer ses besoins en paramétrant le champ DS. Les paquets sortants du coté client sont acheminées vers le fournisseur de service internet. Ce dernier fait passer ces paquets par un module de

classification. De telle façon, le flux des données aura un traitement qui corresponde à la qualité demandée. Il est claire que le client doit communiques ses besoins au préalable avec son fournisseur d'internet. La communication de la qualité de service se fait soit manuellement, soit automatiquement.

Les paquets reçus par le fournisseur sont classifiés et traitées suivant une certain politique. Ces paquets peuvent de même être modifiés. Quand un paquet entre dans un domaine venant d'un autre. Le champ DS peut être remarqué. Grace aux techniques de classification, reformulation et d'ordonnancement ; plusieurs services peuvent être fournis. Le nombre de service possible est réduit par rapport aux autres techniques (MPLS) existantes. Les techniques de classification, d'ordonnancement et de reformulation sont pratiquement implémentées dans les routeurs finaux réseaux (Edges routers). De ce fait, son implémentation est facile.

La bande passante des liens qui lient des clients finaux aux réseaux internet n'est pas de grande capacité (bande passante) comparé aux liens qui forment le réseau internet (entre système autonome).Pour cela, les routeurs ne nécessitent pas des grandes capacités de calcul pour assurer la classification et l'ordonnancement des paquets venants des clients finaux. Les routeurs qui forment le réseau internet et qui n'implémente pas la technique de service différencié appliquent la simple technique de best effort.

(S. Blake et al, 1998) ont défini une architecture scalable pour les services différentiés [30]. Cette architecture assure la scalabilité par agrégation des états de classification de trafic exprimé par le champ DS du paquet IP. Les paquets sont marqués pour recevoir un traitement correspondant. Les politiques de classification et ordonnancement sont implémentées dans les routeurs de frontière (Border Routers). Des ressources sont réservées pour les différents trafics. L'auteur définit un domaine DS comme un ensemble de nœuds DS contigus qui opèrent avec la même politique de provisionnement. Domaine DS a des nœuds DS de frontière qui classifient et conditionnent le trafic entrant. Les nœuds à l'intérieur du domaine DS sélectionnent le comportement adéquat du routage et commutation des paquets. Si des nœuds qui ne supportent pas le DS à l'intérieur de DS, la performance du système devient imprédictible et peut violer le SLA.

Dans un domaine DS, on trouve les nœuds internes et les nœuds de frontière. Les nœuds frontières s'interconnectent avec d'autres nœuds des autres domaines DS qui supportent le DS ou pas. Les nœuds internes s'interconnectent avec d'autres nœuds internes et les nœuds frontières de même domaine DS. Le domaine DS contient des nœuds internes qui implémentent des politiques complexes analogues aux nœuds de frontières. Les nœuds frontières se comportent comme des nœuds DS entrants et sortants. Lors de réception des paquets de l'extérieur de domaine DS, Les nœuds entrants sont responsables de vérifier si les conditions sont respectées du coté de domaines extérieurs. L'architecture définit aussi une notion de région. Une région DS est une région plus vaste qu'un

domaine. Une région DS englobe un ensemble de domaines DS contigus. Les différents domaines DS à l'intérieur de la région DS se mettent d'accord sur les politiques de traitement des paquets entre interfaces des nœuds de frontières de différents domaines DS.

La technique de service différencié est étendue vers un autre domaine DS grâce à un établissement de SLA entre le downstream et upstream de domaine DS. Le conditionnement de trafic performe les opérations de mesure, formulation, remarque pour s'assurer que le trafic entrant respecte les règles de contrat. L'architecture propose deux méthodes de classifications : BA (BehaviourAggregate) agrégation des comportements qui classifie les paquets suivant la valeur du champ DS. Le MF (Multi-field) Multi-champs classifie les paquets à base de plusieurs champs de l'entête comme l'adresse source, destination, champ DS, protocole, port source et destination ainsi que d'autre champs déterminés par le domaine DS. Le problème avec la deuxième méthode de classification (multi-champs) c'est que dans le cas de fragmentation, quelques informations peuvent être perdues et ceci causera un disfonctionnement dans le classificateur.

L'architecture inclue aussi des conditionneurs de trafic. Un conditionneur de trafic peut contenir les éléments de mesure, marquage, reformulation, et éliminateur. En se basant sur une mesure de classification, des décisions sont prises pour annuler, reformuler ou marquer un paquet. Le marquage se fait sur le champ DS. Le marqueur peut être configuré pour marquer avec le même code tous les paquets entrants ou avec une configuration conditionnée par des règles.

Le processus de garantie de service appliqué sur un paquet passe de DS source, vers le nœud frontière du domaine DS, nœud DS intermédiaire et puis le nœud DS de frontières sortant. Avant qu'un paquet sorte du DS source, il est marqué avec un code DS. L'avantage de marquer le trafic au début (source DS) est d'avoir une pré-classification. Ces paquets vont ensuite directement subir l'action appropriée pour chaque type de flux. Le paquet est reçu ensuite par le routeur entrant de domaine DS. Ce dernier doit s'assurer que les règles sont respectées. Il applique toutes les techniques de classification, reformulation et autres suivant la configuration et le code de DS des paquets entrants. Le même processus se répète à l'intérieur de domaine DS. Dans le cas où les paquets traversent des nœuds du domaine non-DS, ces derniers seront traités en best effort.

Pour garantir une certaine qualité de service pour plusieurs flux. Il devait y avoir un ensemble de mécanismes de réservation des ressources, partage des routeurs et des liens d'interconnexion. Le conditionneur de trafic peut gérer l'usage des ressources. Pour la scalabilité, il est préférable d'exécuter les processus de management des ressources à un niveau d'abstraction élevé. Plus le niveau d'abstraction est élevé, plus le protocole

devient scalable et moins en a un contrôle sur les spécifications des trafics. La sécurité reste toujours un aspect majeur dans tous les types des systèmes.

(YoramBernet et al, 1999) a présenté un autre Framework pour les services différenciés [31]. Un trafic traverse plusieurs réseaux interconnectés incluant les hôtes, maison, réseaux de travail, campus et plusieurs réseaux de transition pour atteindre sa destination. Les réseaux de maison et de travail sont des clients de réseaux de campus qui sont des clients des réseaux transitoires. Si une entité souhaite recevoir une qualité de service assurée par le modèle service différencié, tous les routeurs et commutateurs le long du lien (bout en bout) doivent implémenter cette technique. Les aspects techniques entre les deux DS sont définis dans deux types de documents, le SLSs et le TCSs. SLS spécifie les caractéristiques globales et les performances attendues par le client. Les spécifications sont décrites pour l'uplink et le downlink. Les paramètres de SLSs peuvent spécifier le débit, le délai, la probabilité d'annulation d'un paquet et d'autres contraintes et services possibles comme la disponibilité et la fiabilité, le cryptage, l'authentification, la facturation …etc. Les services sont catégorisés comme quantitatifs ou qualitatifs. La demande de qualité de service entre intervenants peut être statique ou dynamique. Un client standard chez un fournisseur de service internet peut simplement demander la bande passante qu'il souhaite. Le client après peut redemander un changement de service. Même chose pour la qualité de service avec les services différenciés où un contrat se fait par l'intervention d'humain (administrateur/responsable) qui définit les différents services et les valeurs choisis. Autre part, la SLS dynamique change fréquemment. Contrairement au SLS statiques, le changement se fait sans l'intervention de l'être humain, ceci nécessite un agent ou un protocole de communication. L'utilisation de service dynamique exige sur les nœuds DS de changer l'équilibrage sur les interfaces fréquemment. Cette tache devient pénible et nécessite des méthodes bien faites que pour ceci. Les applications finales sont aussi concernées par le changement de SLS dynamiquement. Les applications doivent adapter leurs trafics au changement répétitif. Le changement fréquent des SLSs peut causer des problèmes de scalabilité.

Cisco décrit l'ensemble global du concept de DS-QoS et sa configuration dans ses équipements réseaux [32].

2.2.2 Traffic Engineering

Le protocole principal RIP utilisait une technique simple pour le choix des liens en se basant sur le plus court chemin. C'est-à-dire le lien avec le nombre de saut [234]. Les liens peuvent subir une congestion sur toutes les interfaces. Dans ce cas une seule solution consiste à augmenter les capacités des ressources de routage et transferts (medias physiques). Dans l'autre cas, les interfaces diffèrent en termes de surcharge des liens dont des liens seront surchargés alors que d'autres en repos où peu de trafic passe sur eux. L'instabilité des charges sur les différentes interfaces est souvent causée dans le cas d'utilisation des protocoles basant leurs routages sur le plus court chemin comme

RIP, OSPF (Open ShortestPath First) [34], IS-IS (Intermediate System to Intermediate System) [236]. Les liens courts deviennent occupés alors que les liens longs deviennent libres. Une solution alternative du protocole OSPF est l'ECMP (Equal Cost Multi-Path) qui permet le routage sur plusieurs liens qui ont des couts égaux [36] ainsi qu'une alternative de protocole ISIS [37]. Dans le cas où il existe qu'un seul lien court, ECMP n'aura pas d'effet. Parfois, les administrateurs réseaux attribuent des poids aux liens pour propager les flux uniformément. Cette solution devient pénible si le réseau est trop grand et dynamique. Pour les fournisseurs de service internet ça devient impossible.

Le TE (Traffic Engineering) est le processus d'arrangement de flux de trafic dans le réseau pour éviter la congestion de réseau causée par l'utilisation non uniforme des liens de réseau. Le Traffic Engineering complète les services différenciés.

2.2.3 Routage basé sur des contraintes (Constraint Based Routing):

Le routage basé sur des contraintes est un mécanisme de routage complexe dont la sélection de chemin est lié à un ensemble de paramètre QoS exigés par les utilisateurs ou l'administrateur réseau. L'objectif de routage basé sur des contraint est de :

- sélectionner des chemins qui peuvent répondre aux QoS exigées.

- augmenter l'utilisation de réseau.

Pour la sélection d'un chemin, les protocoles de routage basés sur les contraintes viennent en considération non seulement la topologie mais aussi les besoins de chaque flux, les ressources disponibles sur chaque lien et autres contraintes possibles dédiés à des domaines plus spécifiques et parfois déterminés par les administrateurs. Contrairement aux protocoles basés sur le plus court chemin, les protocoles basant sur des contraintes peuvent choisir un lien long et moins surchargé au lieu d'un court chemin surchargé. De cette façon les flux sur les interfaces et liens devient beaucoup plus uniformes. Les entités qui utilisent les protocoles de routage basés sur des contraintes partagent entre elles des informations sur les états des liens. Ces informations sont utilisées pour déterminer le meilleur chemin.

2.2.3.1 Partage des informations :

Les informations sur l'état des liens et les ressources disponibles doivent être échangées périodiquement car ces ressources changent fréquemment. Le terme ressource fait référence à la bande passante des liens. Les protocoles de routages deviennent l'objet d'extension pour inclure ce mécanisme en addition aux différents paramètres échangés [38] [39]. La bande passante des liens change rapidement avec le temps, les entités doivent trouver un compromis entre le nombre de fois les états sont envoyés pour que les résultats soit précises et la non congestion des liens avec beaucoup de trafic de contrôle. Une bonne façon de gérer les liens et de propager les informations importantes uniquement si un changement de topologie ou bien si le changement dans la bande

passante dépasse des seuils précises. Les paquets qui incluent des informations sur l'êtas des ressources ne doivent pas être et qu'il faut pas envoyer à chaque fois qu'un changement intervient, pour cela, Il faut définir un ensemble des timers et des seuils. Si un changement est détecté, il faut attendre que le timer s'alarme pour envoyer les informations. De plus, les seuils vont définir quand on doit considérer le changement d'un paramètre comme considérable [40].

Les métriques sont soit additives, multiplicatives ou concaves. Les métriques utilisées sont le coût, nombre de saut, bande passante et débit, fiabilité, délai et gigue. Les algorithmes peuvent choisir un chemin en se basant sur une ou plusieurs métriques. Le délai, gite, nombre de saut et le coût sont des métriques additives, probabilité de perte de paquet (fiabilité) est multiplicative, alors la bande passante et concave. Le processus de sélection de chemin en se basant sur une multitude de métrique est un problème NP-complet [41]. Les algorithmes qui se base sur le nombre de saut et la bande passante sont beaucoup plus simples [39]. L'algorithme de Bellman-Ford (BF) ou Dijkstra peuvent être utilisés pour calculer le plus court chemin. Peu d'application qui nécessite un délai et une gite. Le délai et la gite peuvent être mappé au nombre de saut et bande passante car ces dernières peuvent définir les deux paramètres de délai et gite. Plusieurs applications de temps réels nécessitent une quantité de bande passante. Le nombre de saut est important et il est important d'associer cette métrique à celle de la bande passante. Tant que le nombre de saut augmente, le nombre de ressources consommées augmentent. C'est pour cette raison qu'on doit minimiser le nombre de saut pour optimiser l'utilisation des ressources dans le réseau.

Les routeurs qui implémentent des protocoles de routage basés sur des contraintes font plus de calcule que les autres protocoles de routage dynamique. Le recalcule de la table de routage se fait fréquemment même si la topologie n'a pas changée. Les calculs peuvent être proactifs ou actives. Les routeurs exécutant des protocoles de routage basés sur des contraintes doivent être dotés de ressources supplémentaires pour satisfaire une bonne qualité de services pour tous les trafics. Les méthodes utilisées pour diminuer la charge des calculs sont :

- Un timer pour réduire les calculs répétitifs.

- choisir la bande passante et nombre de saut comme des contraintes

- utiliser des règles administratives pour repousser certain liens

L'avantage de routage basé sur les contraintes c'est qu'il offre une meilleure QoS pour les flux. Il utilise au maximum les ressources réseaux. Par contre le routage basé sur les contraintes augmente la génération des données sur les medias, la table de routage grandisse, de plus les liens long consomment plus de ressource. Une autre contrainte, c'est que les réseaux dotés de routage basé sur des contraintes, deviennent plus instables que les autres réseaux utilisant des routages standards. Tant que le réseau

grandit, tant que le réseau peut souffrir d'une instabilité. Le déploiement de routage par contrainte doit être fait avec prudence. Sinon, le prix de l'instabilité peut emporter les avantages de ce type de configuration.

2.2.4 Multiprotocol Label Switching - MPLS :

MPSL est un schéma de commutation [42]. Il est situé entre la couche liaison de donnée (Couche 2) et la couche réseau (Couche 3). Chaque paquet MPLS a un entête (Header). L'entête contient une étiquète de 20 bits, trois bit de champ de classe de service (COS : Class Of Service), 1 bit comme indicateur de la pile de l'étiquette et 8 bit pour le champ TTL (Time To Live).

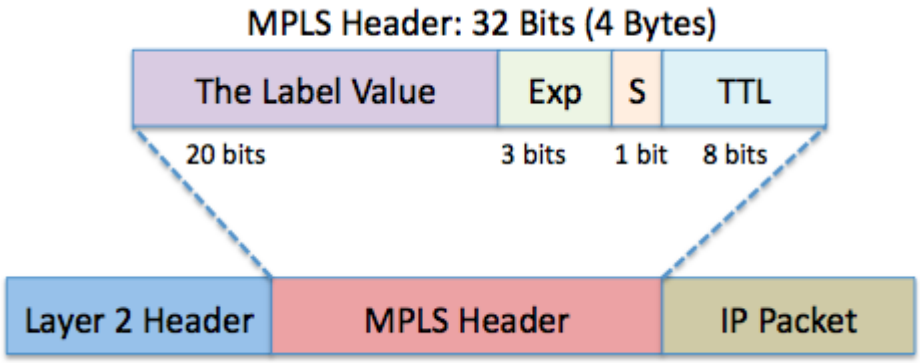

Figure 11: Entete (Header) MPLS; http://blog.ine.com/2010/02/21/the-mpls-forwarding-plane/

L'entête MPLS est encapsulé entre l'entête de la couche liaison et l'entête de la couche réseaux Figure 11. Un routeur qui supporte le MPSL est appelé routeur a commutation d'étiquette (LSR : Label Switching Router). Le protocole de la couche réseau peut être IP ou autres. C'est la raison pourquoi il est appelé multi-protocole. MPLS a besoin d'un protocole pour distribuer les étiquettes pour configurer les LSPs (Label SwitchedPaths). Le protocole RSVP peut être étendu ou bien un nouveau protocole peut être créé pour ce but [43] [44]. Un LSP (Label SwtichedPath) est similaire au circuit virtuel d'ATM (VC). Il est unidirectionnel à partir de l'émetteur au récepteur. Le LSP utilise le protocole pour négocier la sémantique de chaque étiquette par exemple comment traiter un paquet avec une étiquette spécifique par un nœud. La configuration de LSP peut être lancée par un contrôle de trafic comme la mise à jour de la table de routage ou par un type de trafic appartenant à la même classe de service. Le LSP entre deux routeurs peut être le même avec la route de la couche 3 sinon l'expéditeur spécifie une route explicite (ER) pour le LSP. Une table de commutation est indexée par des étiquettes. Cette table est le résultat de distribution des étiquettes. Chaque entrée de la table définit comment traiter le flux portant l'étiquette indexée.

Les paquets sont classifiés et routés à l'entrée du domaine supportant MPLS. Les entêtes sont donc ajoutés. Quand un LSR reçoit un paquet étiqueté, il compare

l'étiquette de l'entête avec l'étiquette d'index des entrés de la table de routage. Ceci est plus rapide que le principe de base qui cherche l'adresse IP avec le masque le plus long [45] [41]. L'étiquette d'entré est remplacée par l'étiquette de sortie et le paquet est commuté vers le LSR suivant. La classification, commutation, et les services de QoS sont déterminés par les étiquettes et le champ COS. Avant que le paquet quitte le domaine MPLS, l'étiquette doit être retirée du paquet. MPLS fournit :

- une classification et une commutation rapide

- un mécanisme efficace de canalisation (tunneling)

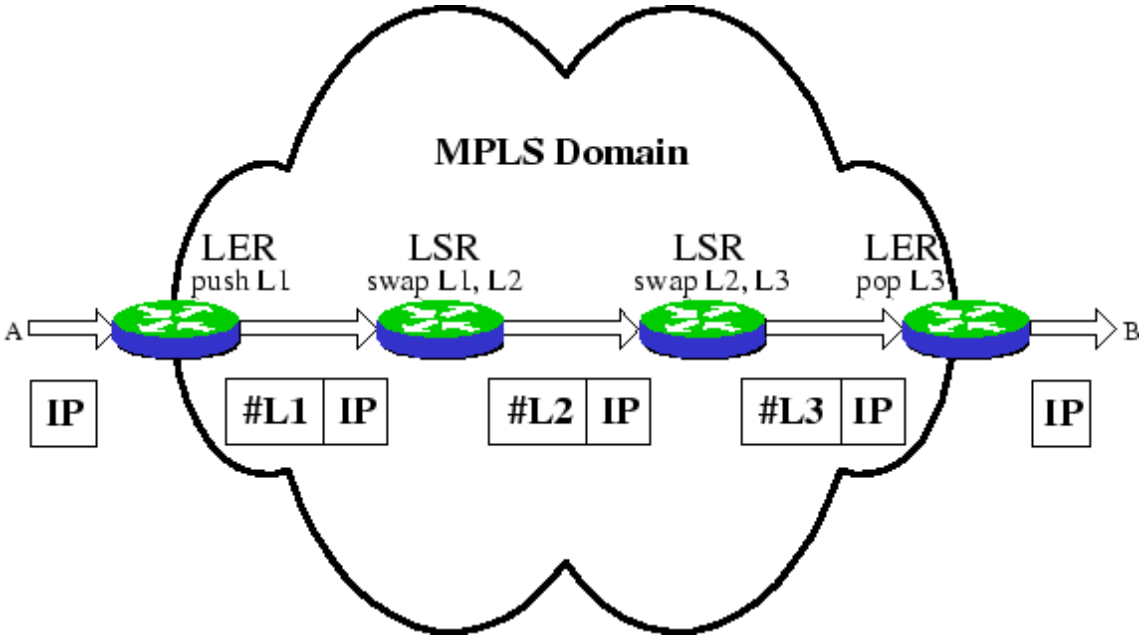

Figure 12 : Traitement des étiquettes dans un domaine MPLS

MPLS remplis les mêmes taches que celui de routage classique mais avec quelques différences :

- la commutation des paquets peut être faite par des commutateurs qui ne supportent pas la couche IP. Contrairement au routage IP, seul les routeurs peuvent commuter les paquets IP.

- les nœuds MPLS peuvent définir des étiquettes indépendantes des informations de la couche 3. Le nœud par exemple attribue des étiquettes différentes au même type de paquet venant de différentes interfaces.

- un paquet qui entre dans le réseau via un routeur particulier peut être étiquetée différemment quand il entre au réseau via un autre routeur. De cette façon, il devient facile de commuter les paquets suivant le routeur d'entré. Ceci ne peut pas être fait dans le routage conventionnel parce que l'identité de routeur d'entrée ne se propage pas avec l'entête de paquet.

- la détermination d'association d'un paquet à un FEC devient compliquée dans certains routeurs sauf que cette complexité n'affecte pas les autres routeurs qui font que la commutation.

- par fois, il est désirable de forcer un paquet pour suivre un chemin particulier qui est choisi explicitement avant ou au moment où le paquet entre dans le réseau au lieu qu'il suive un chemin de routage dynamique. Ceci peut être fait pour suivre une politique ou bien une solution de Traffic Engineering. Avec MPLS, une étiquette peut définir un chemin particulier.

Un protocole de distribution d'étiquette est un protocole qui permet à deux SLRs d'échanger les informations sur l'association Etiquette/FEC que chaque routeur a reçu. L'échange entre deux SLRs peut inclure des informations sur les capacités de chaque SLR. Des extensions de protocole ont été faites pour permettre l'échange des informations comme le protocole MPLS-BGP [47], MPLS-RSVP-Tunneling [48]. Autre part, de nouveaux protocoles ont été créés pour le même objectif dont MPLS-LDP [49] et MPLS-CR-LDP [50]. Un nœud peut demander à un autre des labels d'association pour une classe de trafic spécifique. Ceci est connu sur le nom de distribution de label à la demande. Une autre méthode appelée distribution d'étiquette non sollicité. Les paquets MPLS peuvent contenir plusieurs étiquettes. Chaque étiquette appartient à un niveau d'architecture (voir **Figure 13**).

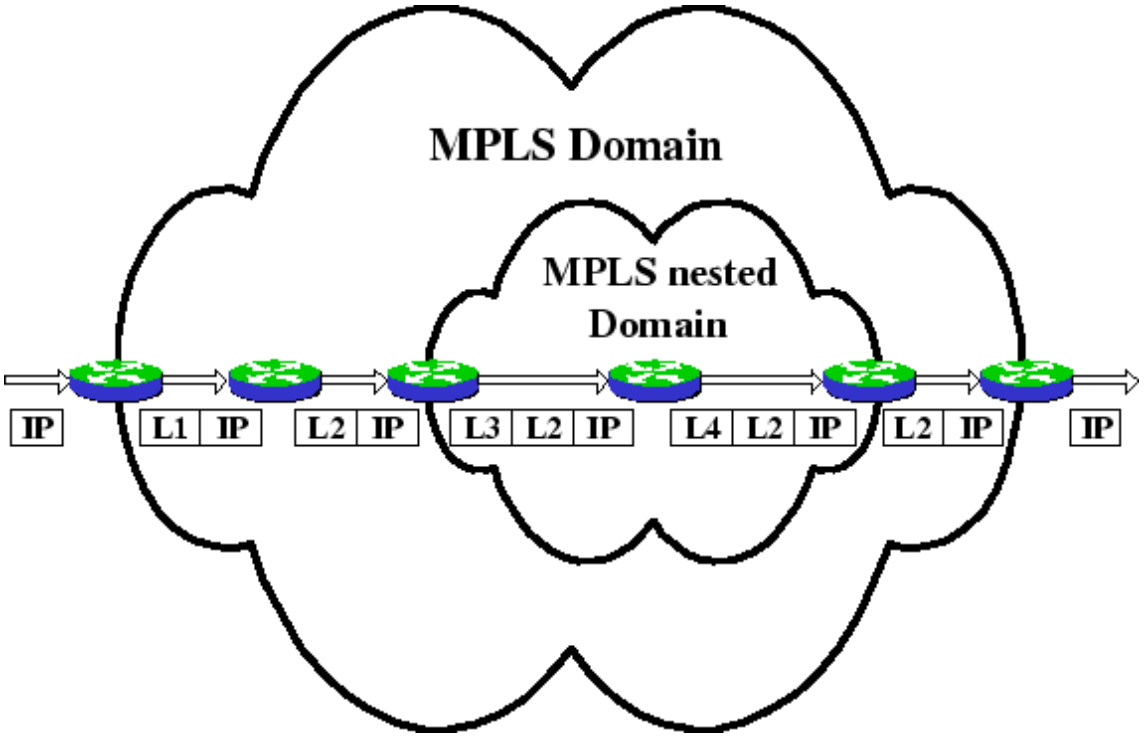

Figure 13 : architecture hiérarchique d'un réseau MPLS avec plusieurs niveaux d'étiquetage ;
http://www.cs.virginia.edu/~mngroup/projects/mpls/documents/thesis/node9.html

Les étiquettes suivent la politique, dernier-entré-premier-servie. À l'arrivée d'un paquet étiqueté avec plusieurs étiquettes, le routeur traite l'étiquète en top. Pour le routage de ce paquet, dans le cas où le routeur suivant a le même niveau que lui, il gardera le même niveau d'étiquetage et il fait passer le paquet. Dans le cas contraire, où le routeur est un niveau inférieur, le routeur va dépiler cette étiquette et faire passer le paquet avec une pille bas niveau. Des canaux peuvent être créés pour chaque lien grâce à la technique d'encapsulation avec des étiquettes multiple. La première étiquette va déterminer le lien principal, par contre la deuxième va définir le canal à l'intérieur de lien défini par l'étiquette une. Les LSRs manipulent une table de commutation dont les entrées sont le saut suivant et l'action à prendre sur le paquet. L'action peut être la suppression, l'insertion ou le remplacement de l'étiquette. Pour minimiser la taille de table de commutation/routage MPLS, les routeurs fusionnent plusieurs étiquettes qui correspondent à plusieurs FECs dans le cas où les FECs suit le même lien et politiques. Donc ces FECs deviennent une seul FEC qui corresponde à une seule entrée dans la table de commutation. De cette façon, la table de routage est optimisée en termes de taille de stockage et temps de processus qui tende vers un petit délai. Dans le cas où un paquet étiqueté ne trouve pas de correspondance dans un routeur MPLS. Le routeur va traiter le paquet suivant les informations de la couche réseau. Comme dans les paquets IP, les étiquettes MPLS contiennent un champ TLL. Le champ est utilisé pour supprimer les boucles. Il est aussi utilisé pour limiter la portée de paquet.

Plusieurs applications peuvent utilisées MPSL architecture. Il peut être utilisé pour définir un chemin bien déterminé à des paquets qui ont une même adresse IP. Comme il est connu, la sélection des routes depuis l'adresse destination d'un paquet IP se repose sur le plus long préfixe dans la table de routage. Pour réduire le temps de recherche, on peut spécifier directement le préfixe. De cette façon, la sélection d'interface de sortie pointe directement vers l'entrée correspondante.

2.2.5 Services intégrés – Protocole de Réservation de Ressource (Resource Reservation Protocol - RSVP)

Le modèle de service intégré propose deux classes de services en addition au service Best Effort [51] [53]. Le service garantie pour les services exigeant un délai limité et un autre service de contrôle de charge pour les applications qui nécessitent un service Best Effort amélioré et fiable [53]. Cette technique permet de réserver des ressources dans le routeur pour assurer une qualité de service pour un flux de client spécifique.

RSVP est un protocole de signalisation pour la réservation des ressources aux applications [54]. L'expéditeur envoie un message PATH vers la destination, ce message est commuté jusqu'au nœud destination. Ce dernier, va répondre avec un message réservation dans le sens inverse de propagation de message PATH. Les routeurs peuvent accepter ou rejeter la demande de réservation des ressources. Dans le cas où un nœud rejette la demande, il initialise un message d'erreur et il l'envoie au récepteur. Dans le cas où tous les nœuds acceptent la réservation, il alloue les

ressources nécessaires comme la bande passante et les buffers pour le flux correspondant. Le nouvel état est sauvegardé Figure 14.

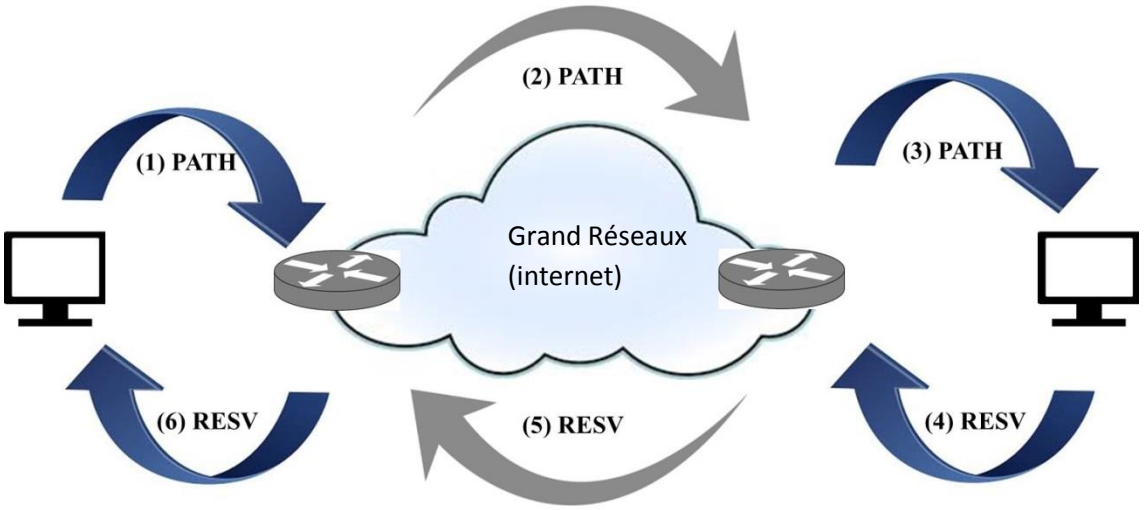

Figure 14 : Signalisation RSVP

Tous les messages RSVP sont encapsulés dans un entête commun qui définit le type et les caractéristiques de message. L'entête contient les champs suivants:

- Version (4 bits) : contient la version de protocole.
- Flags (4 bits)
- Message Type (1 byte): définit le type de message encapsulé au-dessous de l'entête RSVP.

  1 = Message PATH

  2 = Message Réservation

  3 = Message Erreur de Lien (PathErr)

  4 = Message Erreur de réservation (ResvErr)

  5 = Message Destruction de lien (PAthTear)

  6 = Message Destruction de réservation (ResvTear)

  7 = Message de Configuration (ResvConf)

- Checksum (2 Bytes): utilise la méthode de complément à pour la détection des erreurs de transmission
- TTL (1 Byte) : contient la valeur TTL de l'entête IP qui est une valeur qui sera décrémenté à chaque fois qu'elle passe par un routeur. Si elle atteint zero, le paquet sera supprimé.

- Length (2 Bytes) : longueur des données de message encapsulé par l'entre RSVP

La structure de message PATH varie suivant les besoins et les configurations. La forme générale recommandée est comme suit :

- Entête RSVP
- L'Objet intégrité : contient des informations sur la méthode de cryptographie pour l'authentification des nœuds et la vérification d'intégrité des données.
- Session : cet objet contient l'adresse IP de nœud destinataire, le numéro correspond au protocole IP et le port de destination.
- Saut RSVP (RSVP_HOP) : contient l'adresse IP du nœud RSVP générateur de message et l'interface de sortie.
- Valeurs de temps (Time Values): contient le temps de création de message PATH ou réservation. Ce temps est réinitialisé et utilisé pour vérifier combien de temps l'état de session est actualisé ou créé.
- Politique des données (Data Policy): contient des informations sur la politique d'acceptation et rejet des demandes de réservation des ressources localement.
- Template d'émetteur (Sender Template) : cet objet est utilisé pour décrire plus en détails sur l'émetteur.
- Spécification d'émetteur (Sender TSSpec) : contient la définition des caractéristiques de flux d'émetteur.
- AdSpec : contient des informations qui seront exploités et mis à jour par chaque nœud.

L'ensemble des entrés cité ci-dessus ne sont pas nécessairement insérées dans tout message PATH.

Lors le message PATH traverse le lien vers sa destination, il garde le lien inverse pour commuter en retour le message réservation. Le message réservation est construit des entités suivantes :

- Entête
- Intégrité
- Session
- Saut RSVP
- Valeurs temps
- Confirmation Réservation : Ce message est utilisé lorsqu'en demande une confirmation de réservation.
- Portée (Scope) : contient une liste explicite d'adresse IP vers lesquels les informations sont transmises
- Politique des données
- Style : contient la description de la réservation désirée.

- Liste des descripteurs des flux

L'émetteur et le récepteur peuvent tous les deux annuler ou fermer la session de réservation en envoyant un message PathTeardown ou un ResvTearDown. Le message PATH Teardown est initialisé par l'émetteur en cas de timeout. Le message ResvTeardown est envoyé par le récepteur. A la réception de message PATH ou réservation va annuler le processus de négociation des ressources.

Les services intégrés sont implémentés par quatre composants principaux. Le protocole de signalisation comme RSVP. La routine de contrôle d'admission, le classificateur, et enfin l'ordonnanceur des paquets.

Les applications qui nécessitent un service de balancement des charges et le service de garantie doivent définir/configurer le lien et réserver des ressources tout au long de ce lien avant de commencer la transmission des données. Le rôle de composent contrôle d'admission vérifie si une réservation de ressource peut être garantie. Lors de la réception d'un paquet, le classificateur basant sur différents champs met le paquet dans la queue correspondante. En fin, l'ordonnanceur va traiter le paquet dans façon qui assure la QoS demandée pour ce type de flux.

Des travaux comme ceux présentés par (D Ferrari et al, 1990) and (A. Banerjea et al, 1991) ont influencés l'architecture RSVP/Services Intégrés [55] [56]. L'inconvénient avec les services intégrés c'est l'augmentation signifiante de la quantité de stockage des états proportionnellement avec le nombre de flux. Tous les routeurs doivent implémenter RSVP, contrôle d'admission, classificateur et l'ordonnanceur.

Les différents approches et techniques citées peut être combinées dans une façon adéquate au besoin des utilisateurs et l'entreprise. Ces techniques étaient des sujets de recherches dans le contexte d'internet et routage IP. Une autre approche qui était largement étudiée dans des travaux de recherche concernant la garantie et la négociation de niveau de service dans le réseau internet et plus précisément entre les fournisseurs de service internet. Cette approche se base sur le protocole universel utilisé dans le réseau internet BGP-4[57] [58].

2.3 Border Gateway Protocol (BGP-4)

BGP-4 est le protocole de routage utilisé actuellement dans le réseau internet. Tous les fournisseurs de service internet sont obligés d'utiliser ce protocole pour échanger les nouvelles routes crées et supprimées [57]. Le protocole BGP a remplacé l'ancien protocole EGP (Exterior Gateway Protocol) [59].

2.3.1 Architecture de réseau internet :

Le réseau internet est un grand réseau informatique qui connecte un ensemble de réseaux connectés via des routeurs de grande puissance. A un niveau un peu abstrait, nous parlons de connexion entre plusieurs systèmes autonomes (Autonomous Systems).

On peut définir un système autonome comme un ensemble de routeurs sous la même administration. Les routeurs utilisent un protocole interne (IGP : Interior Gateway Protocol) avec des métriques communes pour router les paquets à l'intérieur de système autonome. Un protocole de routage extérieur (EGP : Exterior Gateway Protocol) est utilisé pour router les parquets vers/depuis des systèmes autonomes adjacents. Chaque système autonome a un unique numéro qui sera utilisé comme identifiant durant le processus d'échange d'information. Cet identifiant est codé sur deux Bytes. De plus, il y a une plage d'identifiants qui définit les systèmes autonomes privés et une plage d'identifiant qui définit les systèmes autonomes publiques. Les identifiants sont distribués par Internet Assigned Numbers Authority (IANA) [60]. Les numéros des systèmes autonomes eux aussi souffrent de saturation comme celui de l'IPv4. Les systèmes autonomes sont classés en différentes classes :

- Tier 1 : ce sont les plus grands systèmes qui se connectent entre eux pour former le premier niveau dans la hiérarchie de l'architecture de réseau internet.
- Tier 2 : c'est des systèmes autonomes plus petits que le Tier 1. Les Tier 2 ont des connexions avec Tier 1 pour accéder au réseau internet.
- Tier 3 : c'est des systèmes plus petits que Tier 1 et 2. Elle permet aux utilisateurs finaux de se connecter à internet. ils sont connectés aux réseaux systèmes autonomes Tier 2.
- Transit : un système autonome de transition permet à des clients ou autres systèmes autonomes de router des données via son infrastructure. Dans le cas où le routage n'est pas permis via le système autonome donc il devient un système autonome non transitoire.
- Peering : lorsque deux systèmes autonomes se mettent d'accord d'échanger des trafics entre eux. Le peering se fait souvent entre deux AS de même niveau (Tier2-Tier2 ou Tier3-Tier3).

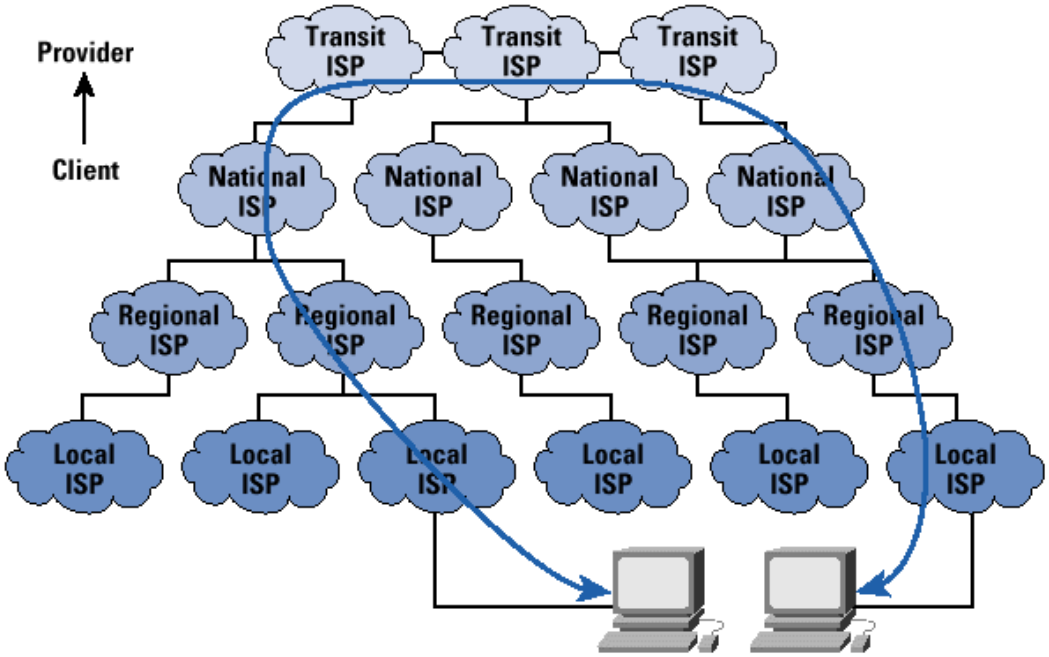

Figure 15 : Architecture présentative d'architecture internet ; http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents/peering-settlements.html

On différencie parfois entre un système autonome de transit et un système autonome peer par rapport à la facturation de service. Le peering est gratuit et les deux systèmes autonomes vont bénéficier de transit des données vers d'autres systèmes autonomes. Si en parle de transit, le système autonome qui offre cette option va facturer ce service pour le système autonome en question Figure 16. Quand un système autonome est connecté avec plus d'une interface au réseau internet on dit que c'est un système autonome multi-homed alors que dans le cas où une seule sortie vers le réseau internet, En parle de système autonome single homed.

La **figure 15** résume l'explication de l'architecture du réseau internet. La figure montre quatre niveau de fournisseur de service internet dont FAI local, FAI régional, FAI nationale et FAI de transite. Le client est situé au-dessous de tous ces niveaux en connectant à un fournisseur d'internet local.

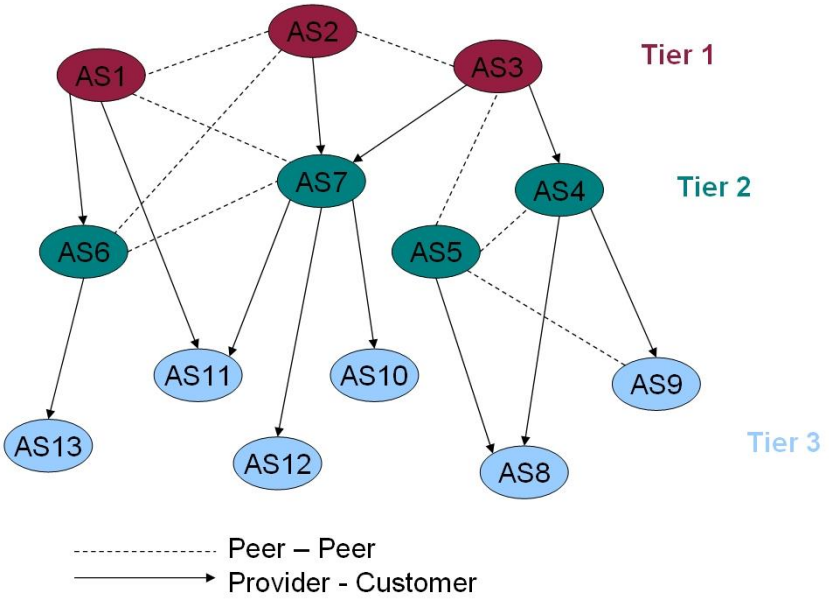

Figure 16 : Transite VS Peering ; https://www.internet-sicherheit.de/forschung/strukturelle-analyse-des-internet/informationen.html

Le routage des flux se fait par rapport à l'adresse IP mais aussi par rapport aux identifiants du système autonome. Donc au lieu de voir l'adresse IP pour connaitre l'interface de sortie correspondante, on peut voir le système autonome qui mène vers cette destination et on fait router le paquet.

Les blocs d'adressages sont distribués par IANA sur les registres internet régionaux (RIR : Regional Internet Registry). Ces groupes sont AfriNIC, APNIC, ARIN, LACNIC et RIP NCC qui sont illustré géographiquement dans la **figure 17**.

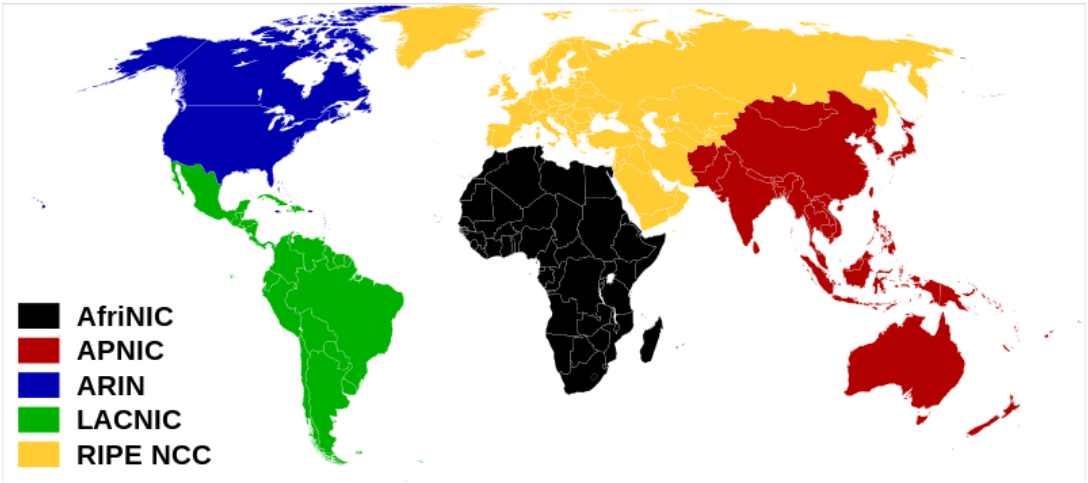

Figure 17 : Registres internet régionaux

Actuellement les systèmes autonomes utilisent le protocole BGP-4 pour le routage dans le réseau internet. Des recherches visant la qualité de service de bout en bout entre

fournisseurs de service internet ont pris le protocole BGP comme modèle de base pour leurs nouvelles solutions proposées. Nous décrivons par la suite des recherches parmi lesquels BGP était utilisé pour apporter la QoS dans le routage du réseau internet.

2.3.2 Description de protocole BGP :

BGP est un protocole d'échange d'information de routage entre les systèmes autonomes. Le protocole décrit des structures de messages, des règles de traitement des messages, des métriques et une machine d'état finit.

2.3.2.1 Les structures de message de contrôle de protocole BGP

2.3.2.1.1Entête :

Comme tout protocole de communication, une structure commune est défini pour qu'elle inclue certaines informations comme la longueur et le type de message encapsulé. Le header/entête de protocole BGP est simple. Il est de taille égale à 19 Bytes. Il est composé de seulement trois champs. Le premier est intitulé Marker, il est de taille égale à 16 Bytes. Le champ Marker est utilisé pour l'authentification entre les communicants. Si ce mécanisme n'est pas utilisé, alors tous les bytes de champ marker seront met à 1. Le deuxième champ est longueur, comme son nom l'indique ; ce champ contient la longueur de message encapsulé. La valeur de longueur de message inclue la taille de header. Le champ longueur est de taille égale à 2 Bytes. Le troisième et dernier champ a la taille égale à 1 Byte et il contient le type de message encapsulé.

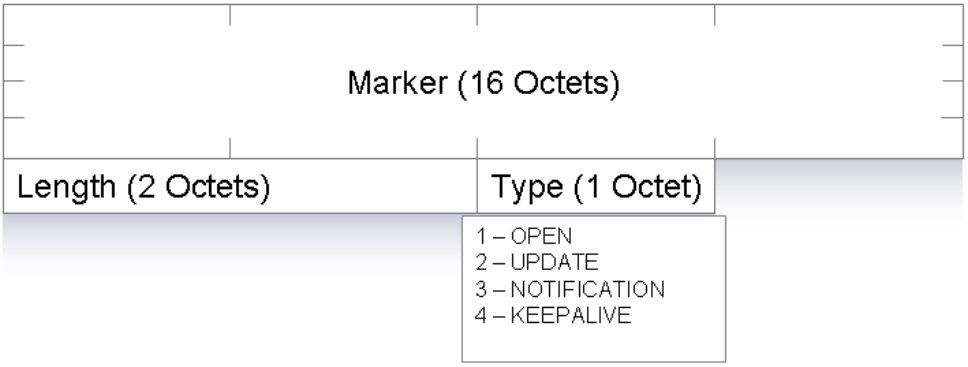

Figure 18 : Structure de l'entête BGP

Le protocole BGP définit 4 principaux messages dont OPEN, UPDATE, NOTIFICATION et KEEPALIVE. Les noms des messages sont significatifs par rapport à leurs rôles. Nous détaillons chaque message dans ce qui suit.

2.3.2.1.2 Message OPEN

Le message OPEN est composé de 6 champs. Il a une taille dynamique. Le premier champ contient la version. Il est de taille égale à 1 Byte. Actuellement, la version 4 est utilisée dans le réseau internet. Le champ suivant est le champ Mon Système Autonome (MyAutonomous System). Comme son nom indique, ce champ contient le numéro de système autonome qui a initialisé ce message. Il est de taille égale à 2 Bytes. Le champ temps de maintien (Hold Time) de taille égale à 2 Bytes contient le temps proposé par l'expéditeur entre deux Message KEEPALIVE ou UPDATE. Le quatrième champ appelé identifiant BGP (BGP identifier), il contient un identifiant de routeur intervenant dans la session. Cet identifiant peut être une adresse IP. L'adresse IP peut être choisie comme la plus grande ou la plus petite des adresses IP des interfaces de routeur. Cet identifiant est utilisé dans toute les interfaces de routeur. Longueur des paramètres Optionnelles (Optional Parameters Length), ce champ contient la valeur de la taille de champ suivant intitulé paramètres Optionnels. La longueur de ce champ est 1 Byte. Le dernier champ a une taille dynamique. Sa taille est déterminée par le champ longueur option.

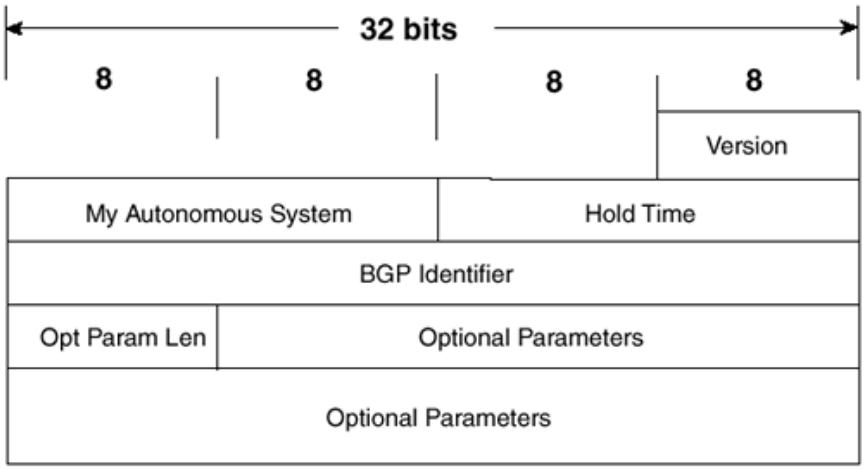

Figure 19 : MESSAGE OPEN - BGP

2.3.2.1.3 Message UPDATE :

Apres l'ouverture d'une session BGP, les deux routeurs communiquants échangent les informations d'accessibilité aux différentes destinations identifiées par des adresses IP. Les informations peuvent contenir des informations des liens vers des nouvelles adresses ou annoncer des liens qui sont devenus obsolètes. Le message UPDATE contient 5 champs. Le premier champ appelée longueur des routes non faisables (Unfeasible Routes Length). Il a une taille égale à deux Bytes. Le nom indique explicitement qu'il contient la valeur indiquant la longueur de champ suivant qui est le champ routes renoncées (Withdrawn Routes et qui contient les adresses IP avec les préfixes (masque) pour informer le nœud adjacent que ces adresses ne sont plus disponibles. Le troisième champ appelé longueur totale des attributs des liens (Total

PathAttributeLength). Ce champ a une taille de 2 Bytes. Le champ suivant est celui des attributs de liens. Les attributs sont des informations liées au lien donnant plus de détails. Le dernier champ est appelé information d'accessibilité de la couche réseau (Network Layer Reachability Information). Sa taille est variable et elle est déterminée en réduisant la taille de paquet indiqué dans l'entête et la taille des 4 champs précédents.

Les champs contenus dans les deux champs withdrawn routes et NLRI contient une liste de doublet (Préfixe, IP). Le champ Attributs contient des paramètres présentés sous forme d'une liste de triplet (Type, longueur, valeur). Le champ Type de chaque attribut est codé sur 1 Byte et il contient deux sous-champs et chacun a une taille de 4 bits. Le premier champ contient des flags et le deuxième contient le code de l'attribut. Le premier flag (bit) définit si l'attribut est bien-connu ou optionnel. Le bit numéro 2 détermine si l'attribut est transitif ou pas. Le bit numéro 3 indique si l'information de l'attribut et partielle ou complète. Le quatrième et dernier bit détermine si la taille de champ longueur est de 1 Byte ou 2 Bytes.

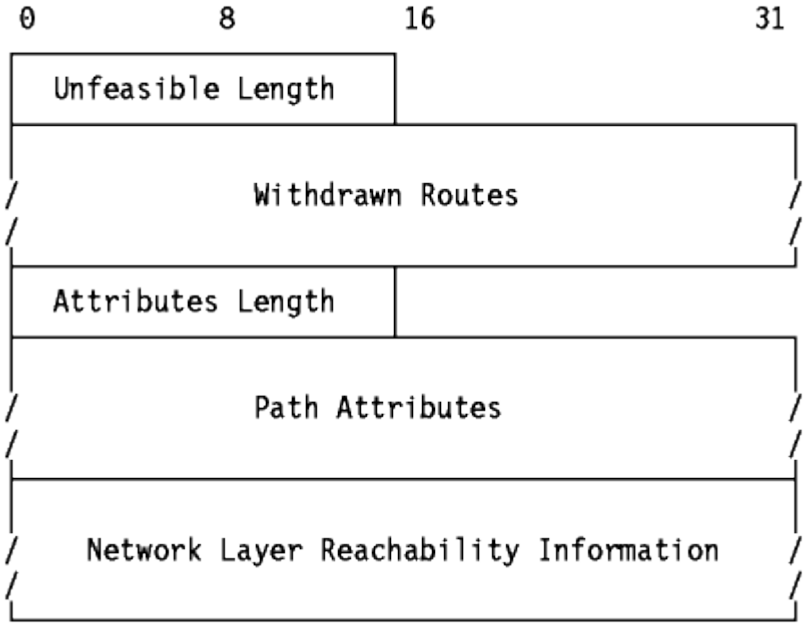

Figure 20 : MESSAGE UPDATE – BGP

Le champ PATH Attributes peuvent contenir des attributs bien définis par le protocole BGP, les Attributs sont les suivants :

- Origin : contribue à la connaissance de l'origine de l'information de lien.
- AS_PATH : contient la liste des identifiants des systèmes autonomes construisant le chemin vers une destination
- Next_HOP : détermine l'interface de sorite.
- MULTI_EXIT_DISC : utilisé dans le cas où il existe plusieurs sorties vers la même destination

- Local_Pref : utilisé à l'intérieur de système autonome pour favoriser une route par rapport à une autre.

2.3.2.1.4 Message NOTIFICATION

Le message notification est simple. Il est envoyé lorsque une erreur est détectée. Ce message est composé de trois champs. Le premier est appelé Code d'erreur. Sa taille est égale 1 Byte. Comme son nom l'indique ce champ contient le code de l'erreur. Le deuxième champ est appelé sous-code d'erreur. Ce champ est utilisé pour spécifier de plus le sous-type de l'erreur. Sa taille est égale à 1 Byte. Le troisième champ appelé donnée est un champ qui contient un ensemble d'information qui peut être sous forme de texte pour donner une information explicite sur le type d'erreur. Ce champ a une taille variable. Les erreurs peuvent être comme ; erreur dans l'entête, erreur de message OPEN, erreur de message UPDATE, erreur d'expiration de timer et erreur de machine d'état fini.

Les sous erreurs peuvent être comme : longueur de message incorrect, mauvais type de message, échec d'authentification, attribut obligatoire non reconnu, boucle de système autonome…etc.

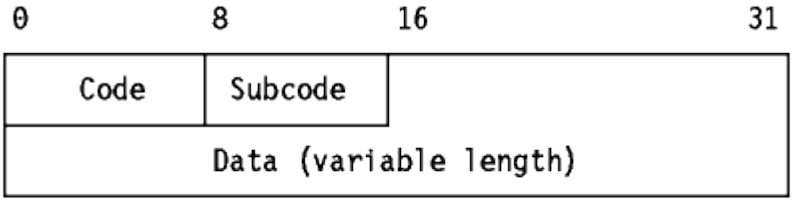

Figure 21 : Message NOTIFICATION – BGP

2.3.2.1.5 Message KEEPALIVE

Le message KEEPALIVE est une structure utilisée pour rafraîchir le timer dans les nœuds communicants. Il est envoyé périodiquement par deux nœuds BGP voisin pour s'assurer de la présence de l'autre nœud. Ce mécanisme est très répandu dans la plupart des protocoles et systèmes distribués. La structure de ce message est la même que l'entête BGP.

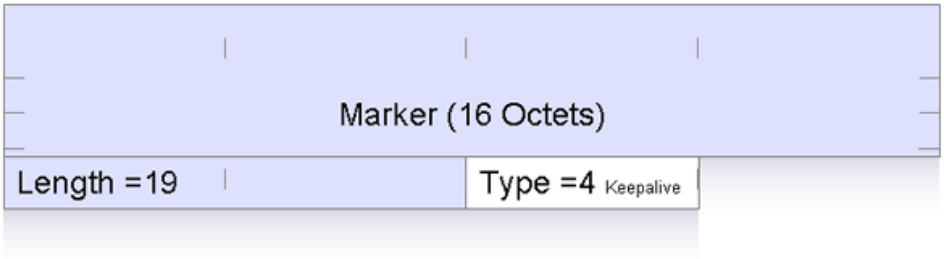

Figure 22 : MESSAGE KEEPALIVE – BGP

2.3.2.2 La Machine d'état fini de BGP

Le protocole BGP définit un ensemble d'état que les nœuds BGP peuvent occuper. La machine définit aussi les transitions possibles entre les états. Les états sont définis comme suite :

- Idle : dans cet état, toutes les tentatives d'ouverture de session BGP de la part du nœud adjacent sont refusées. Lorsque un évènement « Commencer (Start) », le nœud commence à allouer les ressources nécessaires et les timers. Le noeud ouvre unes ocket TCP pour recevoir des connexions d'ouverture de session BGP. Le nœud va changer son état par la suite à CONNECT. Si une erreur est détectée par un nœud (Routeur), les connexions seront fermées et le nœud revient à l'état IDLE. Tout autre évènement que l'évènement START sera ignoré durant cet état.
- Connect : dans cet état, le nœud attend la réalisation de la connexion de la couche transport. Si une connexion est faite avec succès. Le nœud envoie le message OPEN au nœud adjacent et change son état à OpenSent. En cas d'erreur de connexion de la couche transport, le nœud continue à écouter pour d'autres tentatives de connexion. En cas d'erreur, le nœud change son état à Active.
- Active : Dans ce cas, le nœud essaye de connecter au nœud voisin via une connexion TCP (Couche 4 : Transport). En cas de connexion avec succès, le nœud envoie le message OPEN et change son état à OpenSent. En cas d'erreur, le nœud va changer son état à Connect. Si l'adresse IP de nœud souhaitant ouvrir une connexion BGP n'est pas le nœud attendu, le nœud refuse la session et continuer à écouter pour des nouvelles connexions.
- OpenSent : Dans cet état, le nœud attend le message OPEN. Lorsqu'un message OPEN est reçu, le nœud vérifie si tous les champs sont corrects. En cas de détection d'erreur dans l'entête de message OPEN, un message NOTIFICATION sera envoyé et le nœud change son état à IDLE. Si il y'a pas d'erreur, le nœud envoie le message KEEPALIVE et change la valeur de timer avec la nouvelle valeur négociée. Si la valeur dans le champ système autonome est la même que la valeur interne, le nœud estime une connexion interne sinon dans le cas contraire le nœud système estime une connexion externe. L'état change vers OpenConfirm.
- OpemConfirm : dans ce cas, le nœud BGP attend soit le message KEEPALIVE ou NOTIFICATION en cas d'erreur. Si le nœud ne revoie pas le message KEEPALIVE avant l'expiration de timer, le nœud BGP va se mettre dans l'état IDLE avec l'envoie de message notification.
- Established : dans cet état, le nœud BGP peut recevoir les messages KEEPALIVE, UPDATE et NOTIFICATION. Si le nœud revoie le message

NOTIFICATION, alors il change son état à IDLE. Si une erreur est détectée dans le message UPDATE, le nœud passe en état IDLE.

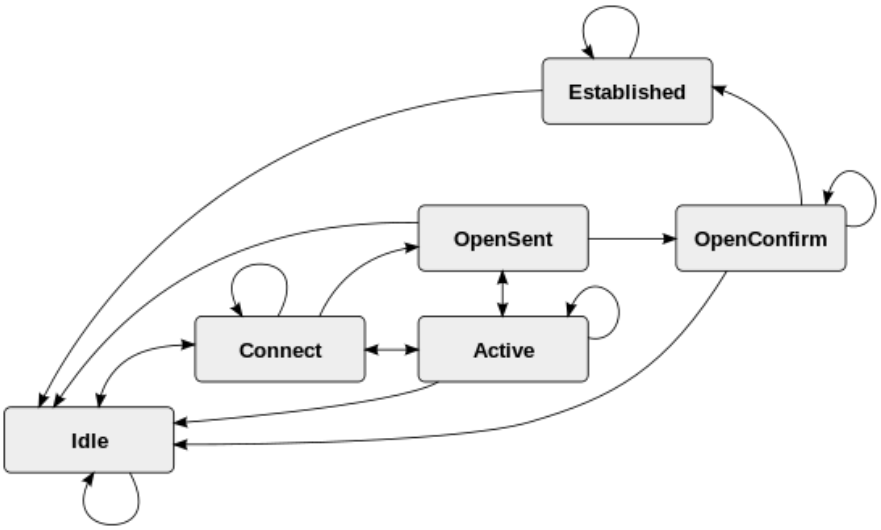

Figure 23: machine d'état fini de BGP

2.3.2.3 Les opérations de protocole BGP :

BGP est un protocole qui a pris sa place dans le réseau internet grâce à sa simplicité. Deux nœuds adjacents se connectent en premier temps avec une connexion TCP. L'utilisation du protocole de communication TCP/IP par BGP lui permet l'implémentation de mécanisme de fragmentation, ordonnancèrent ….etc. La fiabilité de TCP donne un avantage au protocole BGP. Après une connexion TCP/IP, BGP ouvre une session BGP en envoyant un message OPEN. Le nœud met le numéro de système autonome d'appartenance, l'identifiant de nœud lui-même (adresse IP) et la période de timer proposée. Après l'ouverture de session BGP, les deux nœuds échangent entre eux les informations de routage. Les informations sont échangées dans le message UPDATE. Les deux nœuds extraient les informations de routage depuis les messages UPDATE et lance un processus de population de la table de routage. En cas d'erreur durant la session BGP, le nœud BGP envoie un message NOTIFICATION qui inclue le type d'erreur. Durant une session BGP, un timer est utilisé pour s'assurer de la présence de nœud adjacente. A chaque fois qu'un nœud reçoit un message UPDATE ou KEEPALIVE, le nœud remis à nouveau la valeur de timer. En cas d'absence de message UPDATE, le message KEEPALIVE est envoyé explicitement par les nœuds pour informer l'autre nœud de la présence du nœud émetteur.

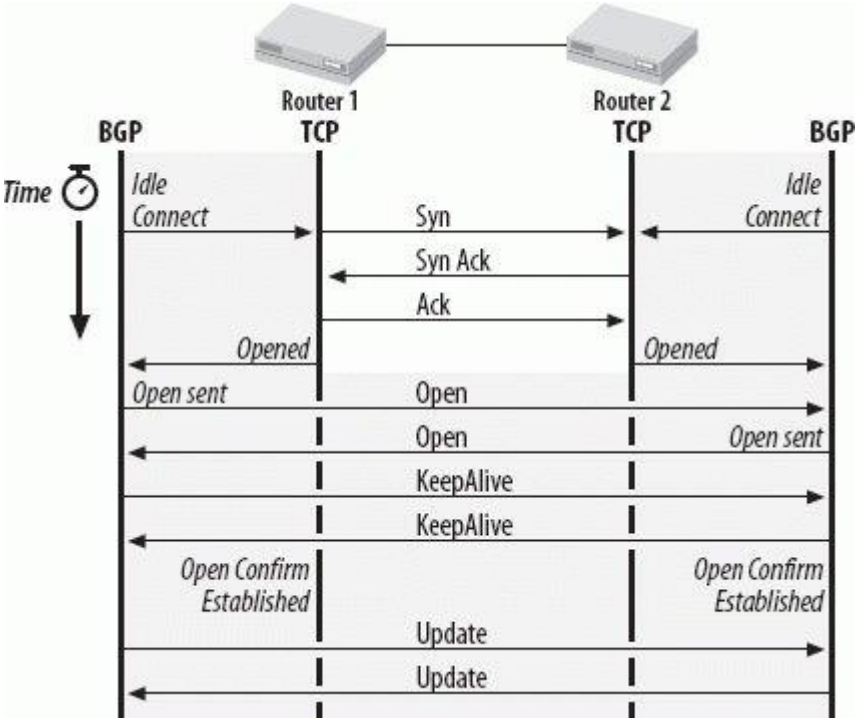

Figure 24 : Diagramme de séquence des opérations de BGP

2.3.2.4 Extension de BGP avec l'inclusion de QoS :

Le protocole BGP-4 ne parlait pas QoS dans ses communications mais son design simple et extensible a permis aux chercheurs de penser à améliorer le protocole pour qu'il traite la QoS dans le choix des chemins. La solution standard propose une définition de nouveaux attributs dans le message UPDATE [61]. Le nouveau attribut est appelé QOS_NLRI. C'est un attribut transitif optionnel. Un lien QoS est un lien qui mène vers une destination avec des besoins exigés. Ces besoins peuvent être exprimés en termes de minimum délai à sens unique, taux de perte des paquets, identification des paquets qui doivent emprunter ce chemin. Les paramètres de QoS peuvent être utilisés comme des entrées de processus de sélection de chemin. La structure proposée est présentée dans la Figure 25.

La structure comporte plusieurs champs dont :

- Code d'information de QoS : contient le type QoS (délai, délai inter-paquet …etc.)
- Sous-code d'information de QoS : contient des informations liées à la QoS demandée dans le champ Code d'information de QoS.
- Valeur de l'information QoS : indique la valeur de QoS
- Origine de l'information QoS : contient le système autonome générateur de cette information QoS.
- Identifient de l'adresse de famille (FAI) : identifie le protocole de la couche réseau.

- Identifiant d'adresse de sous-famille : donne des informations supplémentaires sur les préfix existants dans le champ QoS_NLRI.
- Adresse de saut suivant : contient l'adresse IP de routeur menant vers la destination voulue.
- Flags, identifiant, longueur et préfixe : ce sont des champs qui définissent le NLRI de l'attribut QoS_LNRI.

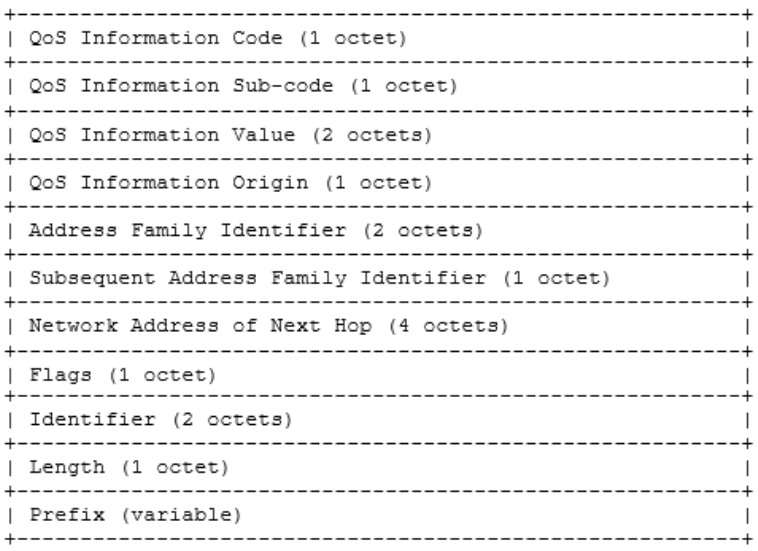

Figure 25: Structure des champs additionnels de QoS-BGP

2.4 La qualité de service dans les Grid Computing:

Le Grid Computing est un modèle des systèmes distribués. Le contrôle des ressources est décentralisé alors que les ressources sont hétérogènes. Le problème le plus répandu dans ce domaine est celui de l'ordonnancement et la distribution des taches sur les machines [62] [63] [64]. L'objectif de Grid est d'achever un système haut débit et de faire correspond aux besoins des applications avec les ressources disponibles. Cet environnement tend à s'intéresser à la qualité de service. La qualité de service dans le Cloud Computing peut aller sur la spécification des besoins, mapping des besoins en une capacité de ressources, négociation de QoS avec d'autres tenants, établir une SLA avec le client, réservation des ressources, gestion des paramètres liées à la session .etc. Dans le cadre des recherches menées sur l'ordonnancement dans le Grid, nous nous présentons les travaux de recherches faites pour apporter d'amélioration au Grid.

(M. Khanli et M. Annaloui, 2006) ont proposé une solution basée sur la base de données active pour assurer la qualité de service dans les systèmes GridComputing [66]. Trois architectures sont possibles. La première, c'est à la ressource de s'annoncer aux clients, ce client décide ensuite à qui il envoie sa demande. Dans la deuxième architecture, le client envoie sa requête à l'ensemble des ressources. C'est à la ressource de décider s'ils peuvent exécuter les requêtes et puis répondre aux clients. La troisième est la

propagation via un broker. Les ressources se présentent au broker et chaque client envoie sa requête au broker. Le broker ensuite décide comment faire correspondre entre les requêtes et les ressources. L'auteur préférait la troisième solution. La solution suit les étapes suivantes :

- les ressources se présentent au broker. Les ressources envoient leurs spécifications (CPU, mémoire, disque dure et composant E/S). Le Broker calcule les autres spécifications (bande passante, délai et taux de perte)

- le broker insère ces informations dans une base de données active

- le client envoie sa requête en spécifiant les besoins QoS

- le broker insère les exigences de client dans une base de données active

- la base de données fait correspondre entre les entrés des informations stockées et l'exigence des clients. S'il n'y a pas une correspondance, la requête doit attendre un certain temps pour ré-exécuter le processus.

- à la fin, le broker envoie les résultats au client

Dans une architecture réseau plus grande, l'auteur préfère de créer plusieurs broker pour former une hiérarchie. Il fait en communication les brokers pour étendre les possibilités de correspondances entre les requêtes et les ressources disponibles.

(V. Talwar et al, 2004) proposent un modèle d'architecture de Grid. La solution est réalisée dans un contexte des sessions de bureau à distance en prenant en considération la performance et la QoS des applications [67]. La clé du design est la structure de requête hiérarchique, modèles des performances des applications, performance de session de bureau à distance et système d'admission de site. L'auteur considère dans son modèle un seul datacenter, le datacenter contient plusieurs nœuds de calcul, serveurs de stockage et un serveur de gestion des ressources. Un utilisateur final envoie une requête pour une session bureautique à distance au serveur de gestion des ressources. Le service de gestion des ressources alloue une fraction des ressources pour les caractéristiques de la charge de travail de l'application. La consommation des ressources par l'application est contrôlée en temps réel. Des techniques statiques sont alors appliquées pour estimer la quantité des ressources nécessaires pour répondre aux besoins QoS. L'exemple pris dans cet article est celui d'ouverture des sessions de bureau distant qui nécessite certaines ressources. Les ressources sont réservées dans les serveurs de datacenter. L'ouverture d'une session de bureau à distance nécessite un serveur d'affichage (ex : X serveurs pour Linux). Le serveur a besoin de mémoire pour s'exécuter ainsi que les applications attachées à son contexte.

Les applications scientifiques ont devenues très nécessiteuses en ressource de calcul et stockage. Les systèmes Grid doivent alors trouver une façon d'intégrer ces contraintes

et en sortir avec un compromis (entre distribution des taches, disponibilité des ressources de stockage et de calcul et la bande passante disponible sur les liens). (S. Charma et Al, 2011) ont proposé une solution décrit comme fiable pour héberger des requêtes de réservation réseau multiple et concurrente entre des sites pairs [68]. L'objectif est de traiter le plus grand nombre de requêtes et de minimiser le temps nécessaire pour le transfert des données. L'auteur a prouvé que ce problème est NP-Complet. Ensuite, une solution heuristique était développée et appelée RRA. L'algorithme RRA s'exécute en itérations. Dans chaque réitération, RRA essaye d'allouer des ressources aux requêtes qui peuvent influencer sur la fonction objective. A la fin il y'aura des requêtes qui n'ont pas été servis. La bande passante disponible sera mise à jour pour l'itération suivante. Dans le cas où aucune requête n'est servie, l'algorithme va s'arrêter et le reste des requêtes ne peuvent pas être satisfait. L'auteur estime que l'algorithme s'exécute en temps polynomial.

Une partie très importante dans l'optimisation des ressources dans les Grid est celle de réplication. La réplication est considérée comme une technique pour l'augmentation d'accès les données et la disponibilité. Les recherche tendent à définir des approches pour déterminer quand et où une réplication doit être ajoutée ou supprimée. Un des challenges de la réplication est la détermination d'un nombre optimal de réplica (ONR) avec garantie de service. De plus, l'endroit où la réplication doit être faite (Lieu) [69]. Des recherches ont été faites sur ce sujet dont celui de (B. Meroufel et G Belalem, 2013). L'auteur a conçu une stratégie de réplication avec optimisation de nombre de réplica avec tolérance aux nœuds défaillants dans la Grid [70].

L'ensemble des problèmes présentés dans le Grid Computing sera plus ou moins les mêmes dans le Cloud Computing. Nous trouvons dans le Cloud Computing comme le Grid Computing le problème de réplication, le problème d'ordonnancement, réservation des ressources, gestion et limitation de bande baissante …etc. pour cela nous nous somme passé par ce concept pour garder une progression explicative de la relation entre les anciennes technologies, le Cloud Computing et la qualité de service point à point.

2.5 La qualité de service de bout en bout dans le Cloud Computing :

Comme toute nouvelle technologie, le Cloud Computing souffre de quelques problèmes comme la sécurité et l'interopérabilité. Malgré ces problèmes, le Cloud gagne de plus en plus de place dans le domaine de nouvelle technologie. Le Cloud est là pour faciliter les choses et donc il a besoin d'effectuer les taches convenablement pour satisfaire ses clients. Concernant la QoS dans le Cloud, des recherches sont menées pour apporter des nouvelles solutions à la garantie de niveau de service à l'intérieur de Cloud. Les chercheurs ne s'arrêtent pas uniquement à la garantie de service à l'intérieur du Cloud mais aussi à la garantie à travers plusieursClouds connectés entre eux. Il est claire que la complexité augmente lorsque on essaie d'implanter la QoS entre les Clouds pour une simple raison c'est que comme décrit, les fournisseurs Cloud développent leurs propres

technologies et ceci cause le problème d'interopérabilité. La question est comment peut-on garantir un niveau de service via plusieurs Clouds si ces derniers ne parlent pas le même langage et n'ont pas les mêmes concepts ?

Notre travail plonge dans cette région de problématique, nous essayons de contribuer à lever l'ambigüité sur la façon de communiquer entre les Clouds pour pouvoir discuter des services avec les besoins qualitatifs des services.

Il est important de consulter les travaux liés à cette problématique. Nous essayerons donc de résumer les travaux les plus proches à notre contribution pour tracer une ligne virtuelle sur les points clés de la qualité de service de bout en bout dans le Cloud Computing.

La sécurité est une qualité de service. Un des challenges de réalisation d'un système multi-Cloud est la sécurité. Comment apporter la sécurité lorsqu'on travaille avec de multiple Clouds ? (O. Incina et al, 2014) a traité ce point et il a proposé une architecture Cloud qui permet une communication sécurisée. L'auteur identifie les entités suivantes : Portal, Service provider, inter-Cloud broker, Cloud fédérations, inter-Cloud exchange, catalogue et service. Le broker reçoit les requêtes et les redirige vers l'inter-Cloud exchange. L'inter-Cloud échange consulte le catalogue pour trouver la meilleure correspondance entre les ressources disponibles et les requêtes. Des exemples de cas sont données comme : demande d'une ressource, rejoindre une fédération, publier des ressources, mise à jour des ressources ou quitter une fédération [71].

Dans un autre travail, (N Grozev et R Burry, 2012) propose dans le cadre de l'inter-Cloud un broker d'application [72]. La solution supporte les capacités suivantes :

- Gestion automatique des ressources à travers plusieurs Clouds pour une application donnée. Ceci peut inclure l'allocation et la de-allocation.
- Un déploiement automatique des composants des applications dans le processus de contrôle
- Ordonnancement et balancement des requêtes entrantes

L'auteur a aussi classifié les inter-Cloud dans quatre classes :

- Fédération volontaire (VolunteerFederation) : c'est lorsque un groupe de Cloud participe volontairement entre eux pour collaborer et échanger les ressources.
- Independent : c'est lorsque plusieurs Clouds sont utilisés en agrégation par une application ou son broker. Dans ce type des applications peuvent utiliser des ressources disponibles sur des Clouds gouvernementaux et des Clouds privés
- Centralisé : l'ensemble des Clouds collaborent via une instance centralisée. L'instance peut être responsable de l'allocation des ressources, registre des

informations partager entre les Clouds pour le processus de sélection des Clouds et construction de service composé fiable.

- Point-a-Point (Peer-to-Peer): dans cette classe, les Clouds négocient des services directement entre eux.

Une contrainte nécessaire pour la réalisation de l'interopérabilité est la création d'un standard. (K. wang et Al, 2012) discute les deux points l'interopérabilité et la standardisation [73]. Le travail introduit en général les concepts nécessaires pour aboutir aux deux points cités. L'auteur donne le XMPP (Extensible Messaging and Presence Protocol) comme une solution favorable dans la communication inter-Cloud (échange des messages) [74]. Par contre il propose un format indépendant des systèmes d'exploitation et les plateformes pour l'échange entre les systèmes de stockage des Clouds intervenants. Le type de format est appelé format des données uniforme (UDF : Uniform Data Format). Un aspect très important dans l'interopérabilité est celui de transfert des machines virtuelles, cette tache nécessite des interfaces uniformes. Un autre travail présenté par (G. Kecskemeti et al, 2012) étudie la gestion autonome des ressources sur un Cloud fédérateur (plusieurs Cloud) [75]. Le système intervient dans des situations comme une défaillance, traitement des pointes et les chutes avec l'aide d'un système de gestion des connaissances. Le système de gestion des connaissances suggère des actions réactives et aussi l'accomplissement entre le SLA établie et la consommation des ressources. L'architecture offre deux options pour l'intégration de système de gestion des connaissances (KM : Knowledge Management). Le système existe en position locale et globale. Le système local est implanté par déploiement de chaque Cloud. Chaque Broker d'un Cloud utilise un KM séparé pour ses besoins internes.

(MM Hasen et al, 2012) revient sur la collaboration inter-Cloud avec une approche de collaboration entre service Cloud appellée Collaboration Cloud Dynamique (DCC) [76]. Les fournisseurs Cloud grands, moyens et petits se complètent pour gagner une grandeur et élargir leurs capacités pour répondre aux besoins QoS. L'auteur décrit une architecture dont les composants, caractéristiques et cas d'utilisation. Les étapes de formation de DCC sont les suivantes :

- Un pCP (initiateur) trouve une opportunité à partir de répertoire des informations ensuite il trouve un partenaire adéquat en utilisant le contrôleur de sélection de collaborateur qui fournit une liste de collaborateurs possibles.
- Après la sélection d'une combinaison de partenaires, le médiateur obtient le service ou les ressources et accède à l'information à partir du répertoire des services, les politiques et les SLA à partir de répertoire des politiques, Génère un eContract [77] ensuite le EContact est passé à l'agent collaborateur local.
- La dernière étape, c'est la négociation de CA local de pCP en utilisant l'eContact. Si tous les CPs se mettre d'accord, ils font un agrément initiale

entre eux. Lorsque pCB acquit tous les ressources et services demandés pour faire face à la QoS demandée, à ce point un nouveau DC devient opérationnel.

Un travail de (ShigetoshiYokoyama et al, 2014) proposaitt une autre architecture pour traiter un problème différent [78]. Le contexte reste toujours l'apport de la QoS dans un environnement Cloud point à point. L'architecture nommée architecture inter-Cloud académique (AIC : Academic inter-Cloud Architecture).Il différencie bien deux situations. La première c'est l'optimisation de communication entre les machines virtuelles qui résident dans le même lieu (datacenter). Alors dans ce cas, un routeur virtuel s'occupe de routage de trafic et communication entre les machines virtuelles. La deuxième situation où le problème exige plus de travail pour optimiser la communication en termes de latence entre des machines virtuelles qui se trouvent dans différentes architectures réseaux. L'architecture comporte deux composants de service ; le premier est AIC de calcul (Compute) et le deuxième est AIC stockage (Storage). Le composant service AIC compute permet aux utilisateurs de créer un Cluster de serveurs physiques et par la suite déploiera des applications et logiciels pour construire des IaaS et PaaS. Le composant service Storage permet aux utilisateurs de stocker des données comme des machines virtuelles. Il considère que les Clouds sont connectés avec des liens de hautes capacités avec possibilité d'utiliser la technologie des VPNs.

(Walter Cerroni, 2014) fournit un model analytique pour dimensionner les capacités des réseaux partagés et les serveurs de stockage dans un Cloud fédérateur [79]. Le travail exercé se repose sur des hypothèses. L'auteur assume que le réseau fédérateur est composé de plusieurs datacenters interconnectés à distance par un réseau maillé avec des pipes de bande passante garanties. Les pipes peuvent être réalisées avec la technologie MPLS. Le trafic généré dans le réseau est assumé être généré uniquement par les machines virtuelles. Le modèle choisit les ressources destinataires pour une migration d'une machine virtuelle en se basant sur le débit de lien menant vers la machine physique destinatrice qui soit supérieur au débit exigée pour migrer une machine virtuelle z. de plus, la ressource demandée ne doit pas être occupée par une autre machine virtuelle. Le modèle s'intéresse beaucoup plus sur la migration des machines virtuelles (mémoire).

Pour permettre la communication et l'interopérabilité entre différents fournisseurs Clouds. Il est important de; soit définir un standard de chaque service et que tous les fournisseurs doivent le suivre. Chose qui est trop loin d'être réaliste. La deuxième solution c'est la définition des interfaces standard et que chaque fournisseur Cloud implémente ces interfaces à sa façon. Les Clouds peuvent demander des services avec l'exigence d'un certain niveau de service en fournissant les paramètres des interfaces standard. La demande des services nécessite un processus autonome conditionné par des évènements spéciaux comme par exemple insuffisance des ressources dans le Cloud local, migration d'une machine virtuelle vers un autre fournisseur Cloud pour

augmenter la disponibilité et optimiser le temps de réponse par rapport au changement de lieu d'accès de l'utilisateur. En plus de la définition des interfaces des services, le système nécessite aussi alors des protocoles de communications. Ces derniers définissent des structures de message et des règles de traitement.

(Muhammad Bilal Amin et al, 2012) propose un middleware basé sur publier-inscrire (publish-subscribe) pour l'échange des messages entre Clouds [80]. L'auteur implémente le service de distribution des données (DDS : Data Distribution Service) avec un modèle d'information qui est supposé devenir une solution fiable pour le domaine d'échange des messages du standard d'interopérabilité inter-Cloud. L'architecture ne s'oppose pas aux autres. Elle ajoute aux opérations basics celle de publication et inscription des objets. Le modèle d'information définit l'entité requête pour la découverte d'un Cloud interopérable et réponse par un Cloud de même domaine. Ces deux entités ne sont pas confusées avec requête et réponse standard. Le modèle définit ce qu'on appelle requête concrète et réponse concrète. Le message inclue une définition de type avec des énumérations qui permettent de classifier les messages comme requête de migration de ressource (MSG_RMR) ou un requête précédemment envoyée (MSG_DSP) qui indique que la requête précédente envoyée est invalide et tous traitement à propos de ça doit être arrêter ou annuler. Une autre information importante est les paramètres de QoS. La politique de QoS définit les paramètres QoS par publication ou inscription. Le troisième paramètre c'est le descripteur des ressources, il donne des spécifications sur le Cloud. Il est basé sur le langage web d'ontologie OWL. Malheureusement le travail présenté ne détail pas des structures de message pour la requête et réponse …etc. sauf qu'il donne des vues d'ensembles sur le fonctionnement de modèle. Il liste les paramètres de qualité de service :

- Date limite (Deadline) : temps maximum entre l'arrivé de deux messages.
- Durabilité :
- Budget de latence (Latency budget) : temps permis pour délivrer un message.
- Durée de vie (lifespan) : duration lors le message est valide.
- Fiabilité : si il nécessaire de transférer le message avec fiabilité ou non.
- Vivacité (liveliness) : un mécanisme pour détecter si un nœud racine est vivant ou mort.
- Priorité de transport : pour la classification de message par priorité.

Comme le modèle se base sur une ontologie, l'auteur a décrit l'ensemble des domaines et rangs de la description des ressources de Cloud (Vendeur, info, nom, capacité, id, OS, ram, version).

Dans un travail similaire à celui de (Muhammad Bilal Amine et al, 2012), un protocole de messagerie inter-Cloud était aussi proposé. Pareil, le travail ne contient pas une

présentation des messages de contrôle ou de communication. (Yonggang Wen, 2011)[81]. Le protocole est structuré en trois composants qui sont véhicules, format de message et contenu message. L'auteur essaye de donner quelques options de composants véhicules et format de message. Il catégorise le message dans six parties. Première chose, l'auteur identifie les six classes de contenu :

- Distribution de contenu : un message pour déplacer le contenu de la source aux utilisateurs
- Requête de routage : un message pour naviguer les requêtes des utilisateurs vers l'endroit adéquat pour retrouver le contenu demandé.
- Adaptation de contenu (content rendering) : message pour la création ou l'adaptation de contenu pour convenir les performances de l'utilisateur et les capacités des équipements.
- Gestion des ressources : un message pour le contrôle, allocation et réservation des ressources.
- Autorisation, authentification et comptabilité : un message pour permettre le contrôle, enregistrement, comptabilité et la facturation de l'utilisation de contenu.
- Divers (Misc) : message pour des extensions possibles.

Dans le deuxième module, le message est codé dans différent format. Le format de message fait référence à la méthode d'encodage de message pour qu'il soit échangé par plusieurs Clouds. L'auteur présente trois types de formatage dont :

- Type-longueur-Valeur : avec ce formatage, type contient de code qui présente la catégorie de contenu. Longueur contient la taille de champs qui contiendra la valeur de la catégorie de donnée. Le champ valeur contient bien la valeur liée à la catégorie de l'information. L'ensemble des données sont présentées en format binaire.
- Champ-valeur : dans ce type de formatage, les données sont encodées sous forme de texte. Ce type de format est utilisé dans les protocoles basés sur le TCP/IP comme http, FTP, SIP, POP3 et SMTP [82].
- XML : dans ce type de formatage, XML est utilisé pour implémenter des messages entre différents nœuds dans un réseau Cloud.

Le choix de l'auteur tombe sur le premier type de formatage TLV (Text-Length-Value). L'auteur justifie son choix avec la rapidité de parsing des messages codés en TLV comparant par exemple à celui de XML.

Un dernier élément du modèle est celui de véhicule de message qui est une sort de canal ou media de transfert des messages entres Clouds. Le design inclue les propositions suivantes :

- RPC : dans ce type de véhicule, chaque Cloud est considéré avoir un serveur exécutant un processus d'échange. Les messages seront donc échangés àtravers les « remoteprocedural calls »
- Web service : tous les Clouds offrent un nœud de web service pour l'échange des messages. Les messages peuvent être pris en charge par des protocoles standards comme SOAP [84] en définissant les interfaces dans un fichier WSDL [83] qui permet à chaque Cloud d'implémenter la logique des interfaces.
- Mise en queue des messages (Message Queuing) : dans ce design en partage un bus avec tous les nœuds participants (Clouds). Un exemple de ce modèle est le XMPP.

L'auteur préfère utiliser la dernière option pour une raison que le XMPP est largement utilisée dans le déploiement des réseaux actuels et qu'il est considéré comme candidat pour la communication inter-Cloud. De plus avec XMPP, en peut utiliser une seule base de donnée pour les contenus, le traitement et les utilisateurs.

Malgré l'absence de définition des structures exactes sur les messages et les méthodes de transfert et de communication dans le travail proposé mais il reste très intéressant car il regroupe dans un travail bien résumé l'ensemble des points basiques importantes pour la réalisation d'un protocole de communication pour la qualité de service dans un environnement multi-Cloud.

Le travail où nous nous sommes trouvés proche par rapport à notre travail était celui de (WassimItani et al, 2014) [85]. Ce travail à bien était un protocole dans le sens propre du terme. On trouvera des définitions des structures des messages et des règles de traitement avec le processus de sélection. L'idée de l'auteur est de transformer le protocole BGP en nouveau protocole qui cherche des services aux lieux de chercher des adresses IP. L'idée est simple, il prenait des messages de protocole BGP et il changea les termes de champs en lui adaptant par rapport au numéro de service et type de QoS. Nous présentons plus en détails ce travail pour raison de l'axe commun pris pour lever le problème communication d'inter-Cloud. Le protocole ServBGP est donc un protocole de routage de service pour la gestion de collaboration des services des fournisseurs Clouds. Il est basé sur le protocole BGP. Le protocole gère les différents aspects d'interaction et collaboration entre services et Clouds comme la découverte, annonce, consommation et libération des ressources. Le processus de sélection des liens et services prend en charge le prix de service et la qualité de services. L'auteur a changé les noms des champs pour servir au routage service inter-Cloud. L'architecture d'hypothèse est composée de client demandeur de service. Le service peut être un logiciel, stockage, plateforme ou des ressources infrastructures. Un fournisseur de service Cloud (CSP : Cloud Service Provider) qui répond directement, indirectement, complétement ou partiellement au service des clients. Le modèle dans ce travail classifie les fournisseurs de service Cloud en plusieurs types Figure 26.

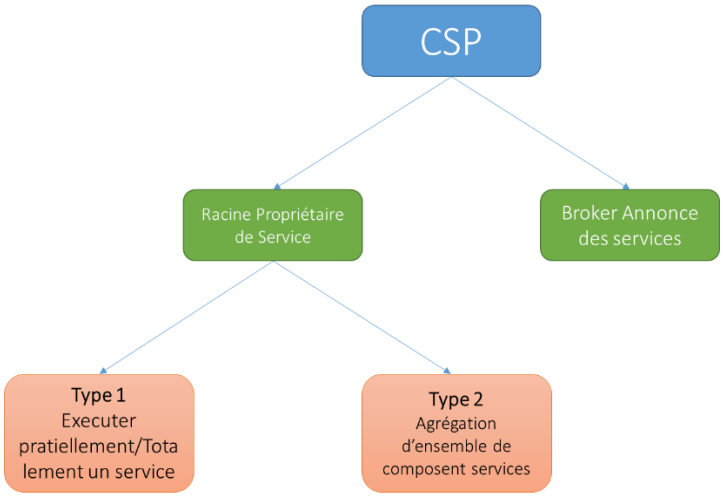

Figure 26 : Classification des Fournisseurs de service Cloud

- CSP racine : ce type de fournisseur possède un service particulier.
- CPS complet (Full CSP): ce type de fournisseur exécute la totalité des services dans l'espace de son infrastructure.
- CSP partiel : contrairement au CSP complet, il a un service dans son espace mais collabore avec d'autres Clouds pour leurs fournir des services complémentaires à son service principale.
- CSP type 2 : ce type de fournisseur Cloud possède un service sans exécuter aucun module dans son espace. En SOA, ceci est connue sur le nom de composition.
- CSP Broker : c'est un agent autorisé pour annoncer des services sans participer à l'exécution de service.

Un exemple très concret était donné pour bien distinguer entre ces types de CSPs. Dans la Figure 27, P5 de type 2 publie l'offre d'un service de gestionnaire de base de données relationnels qui est en réalité proposé par P4. Ce dernier lui-même agrégeât les trois services offerts par trois différents CSP (P1, P2 et P3) et ajoute à ces derniers sa logique d'exécution sur son infrastructure. P6 est un broker qui permet au CSP P5 de publier et annoncer ses services à d'autres nœuds.

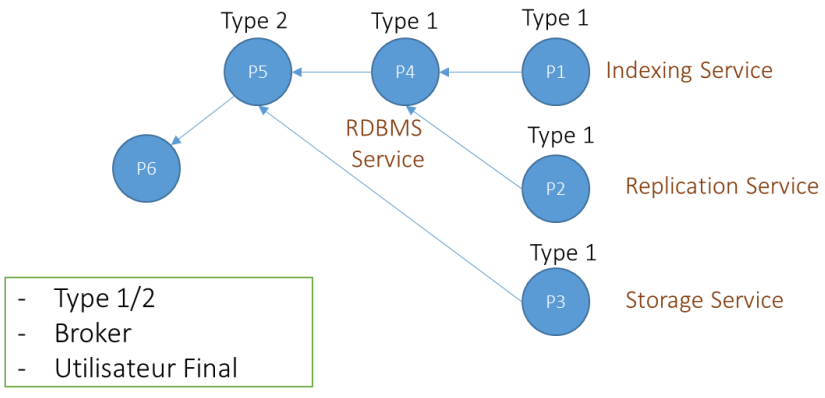

Figure 27 : Simple Scenario de Collaboration des différents types de CSP

De plus que l'architecture proposée, le protocole introduit un nouvel élément qui est le message d'annonce (Advertisement). Ce message est associé avec l'opération de recherche de lien. Le rôle de routeur qui exécute le protocole ServBGP et de faire commuter les requêtes des utilisateurs vers la destination appropriée en se basant sur le cout de service. En autre terme, le routeur ServBGP exécute un processus de sélection qui fournira le meilleur chemin vers la destination pour assurer la QoS demandée. Les fournisseurs de service Cloud annoncent leurs services aux autres via un message d'annonce de service (SAM : Service Avertissement Message). L'auteur décrit un et un seul message de communication pour son protocole. Le message présenté dans Figure 28 contient :

- ID de Fournisseur annonceur : contient l'identifiant de fournisseur Cloud qui a envoyé le message.
- Destination ID : il contient l'identification de fournisseur Cloud destinataire.
- Nombre des messages d'update : le message SAM contient une liste de sous-message de mise à jour qui ont la même structure sauf que chaque message contient un type de service identifié par un identifiant de service.
- Le sous message contient les champs suivants :
  - ID Service (SID): c'est le numéro de service unique par CSP. Le doublet (PID, SID) sera donc un identifiant global par rapport à la totalité de l'infrastructure.

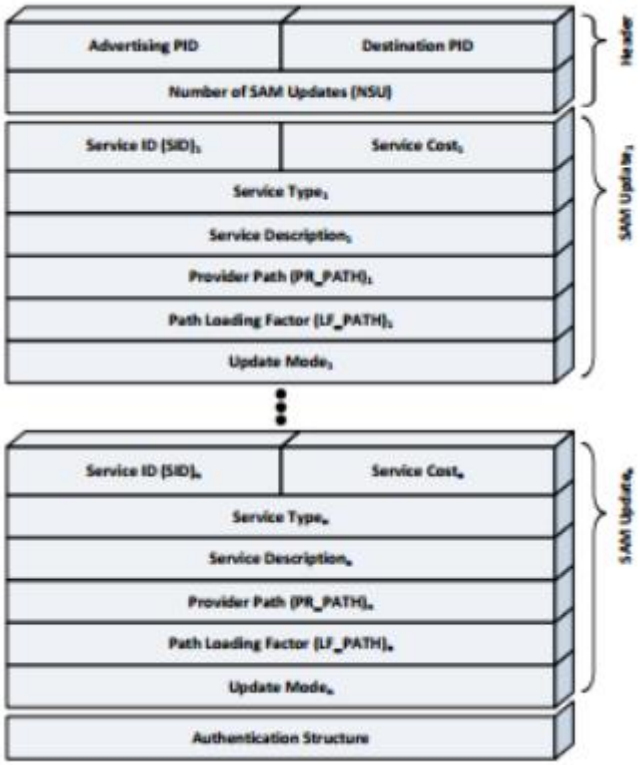

Figure 28 : Message d'annone des services

- Cout de service : Le fournisseur d'un service annonce le service avec le cout pour la facturation de service
- Type de service : ce champ détermine le type de service (SaaS, PaaS, IaaS …etc.)
- Description de service : contient des métadonnées décrivant les opérations de service
- Lien vers le fournisseur (PR_Path) : come BGP, il contient la liste des fournisseurs Clouds intermédiaires qui mènent vers le fournisseur Cloud destinataire. Initialement lorsque le message SAM est créé, ce champ contient uniquement le PID de l'annonceur.
- Facteur de charge de lien : ce champ enregistre une séquence de valeur en pourcentage. Il indique la charge sur les ressources dans les infrastructures Cloud intermédiaire (CPU, Mémoire, bande passante…etc.).
- Mode de mise à jour : ce champ indique si ce message annonce une disponibilité de service ou la révocation.
- Structure d'authentification : ce champ vient à la fin de la liste des SAMs concaténés, il contient une signature numérique sur toute la liste des messages SAMs.

Les messages traversent les nœuds et ces derniers vont extraire et parser les informations contenues dans le message et les sauvegarder dans des structures internes

qui seront utilisées par la suite dans le processus de sélection de lien parmi plusieurs choix tous dépends des exigences de client et les liens disponibles. La chaine de parsing des informations annoncées dans le message SAM suit le schéma illustré dans la Figure 29.

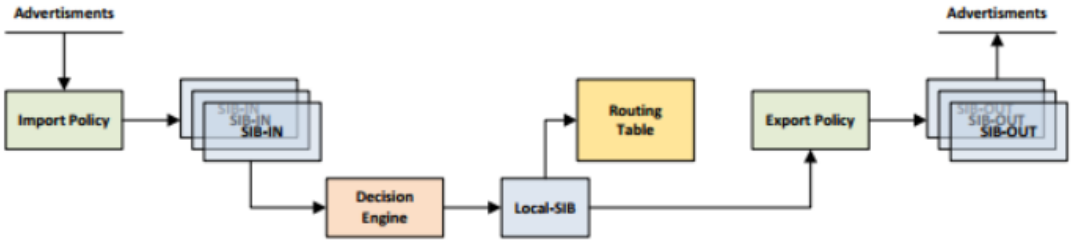

Figure 29 : Etapes de sélection des liens

En comparant la chaine de sélection des liens en trouve une très forte similitude entre les entités avec le processus de sélection dans le protocole BGP ce qui est confirmé par l'auteur. L'auteur partage le processus en deux partis. La première concerne la sélection des liens et la deuxième concerne l'annonce des services. De même que les trois structures de table de routage dans BGP, alors les mêmes existent dans le protocole proposé (SIB-IN, SIB-LOCAL, SIB-OUT). Quand un CSP reçoit un message SAM, il parse les données et enregistre les informations dans la première structure qui est le SIB-IN. Plusieurs liens peuvent mener vers le même CSP et plusieurs fournisseurs peuvent fournir le même service. Tous à fait correcte que l'auteur indique que non pas tous les informations inclues dans le message SAM vont être enregistrées dans SIB-IN car ce processus suivra des règles et des politiques de filtrage. De même pour la structure SIB-OUT, c'est-à-dire que il y'a aussi des règles et des politiques qui déterminent quel données doivent être annoncée au Routeur/Nœud/Fournisseur Cloud voisin. La base de données locale (SIB-Local) garde les meilleurs premiers et deuxièmes choix sélectionnées par le processus de décision. La structure SIB-LOCAL détermine explicitement le contenu de la table de routage principale. Le format de chaque entrée dans la table de routage est définie par le triplet SID, Type de service, PID de prochain saut. Les trois structures sont similaires dans la structure qui est pareil au protocole de BGP dont l'idée était prise depuis ce dernier.

2.6 Conclusion

Le problème de la qualité de service dans l'environnement Cloud a une liaison à des technologies précédentes ; GridsComputing, les Clusters, internet et autres. Le Cloud Computing traite la qualité de service dans des cas particuliers. En traitant la qualité de service entre fournisseurs Cloud, nous devons définir des standards et des interfaces communes pour permettre à différents fournisseurs Clouds de communiquer. Les technologies succèdent et chaque technologie devient un support à l'autre. Des protocoles, des plateformes et des standards ont été créés comme le protocole BGP-4

pour le routage internet, Web Service et protocoles liés étaient développés pour l'architecture orienté service. Le protocole BGP était étendu pour acquérir la possibilité de traitement de QoS point-à-point (entre fournisseurs de service internet). MPLS vient après que des solutions connaissaient des limites de définition d'un grand nombre de classes. Depuis que BGP-4 était développé pour le routage dans internet, il passa vers l'utilisation de la QoS et puis ce dernier était adapté à l'environnement Cloud. Les travaux qui décrivent des protocoles en termes de paquets et règles font rares. Nous avons enrichi cette particularité de type de solution concernant la qualité de service inter-Cloud avec un nouveau protocole qui prend en charge les caractéristiques de Cloud. Le protocole définit des nouvelles structures contrairement aux travaux existants qui se basent sur des protocoles existants comme BGP.

Chapitre 3

Contribution : Un nouveau protocole de routage de garantie de niveau de service de bout en bout dans un environnement Cloud

3.1 Introduction :

Si la plupart des travaux de recherches se sont intéressés à la définition des architectures standard, des brokers et des middlewares, rares sont les travaux qui définissent un protocole complet de communication qui prend en charge les caractéristiques de Cloud. le concept de collaboration de plusieurs Clouds commence à intéresser les industries, les académies et les universités. Le monde se lance dans une approche d'internet des choses. Ce terme signifie que toute chose dans ce monde devient intelligente et quelle peut échanger et communiquer. La multiplicité des composants intelligents augmente la quantité des données échangées. Ces données doivent être traitées et stockées aussi. Avec l'apparition de Cloud Computing, plusieurs compagnies et sociétés sont soulagés car ils se mêlent plus de la partie IT dont le maintien de réseau informatique, le préinvestissement dans les ressource de stockage, de calcul, les logiciels chères, les licences, les serveurs, les Datacenters …etc. Le point fort de Cloud Computing revient à son modèle économique basé sur le payement à l'usage. Le fournisseur de service Cloud doit répondre à tous les besoins des utilisateurs avec qui un agrément de niveau de service signée entre le fournisseur de service Cloud et le client. Le Cloud fait en sorte que les ressources sont infinies et que le client peut demander des ressources autant qu'il a besoin et une fois l'utilisateur n'a plus besoin de ces ressources, il peut libérer alors ces derniers. Pour que le client se libère des coûts de maintenance de l'infrastructure interne et juste d'exploiter le service offert par son fournisseur, Le fournisseur doit s'assurer que toutes les requêtes sont satisfaites. Satisfaire les clients peut être seulement si la ressource disponible dans l'infrastructure répond aux taux des demandes des utilisateurs. Dans le cas contraire, le fournisseur va forcément violer le SLA. Pour lever ce problème, les fournisseurs et les chercheurs ont commencé à investiguer sur une solution possible. Les chercheurs ont une idée commune ; développer des interfaces, des Frameworks et des protocoles standards pour permettre aux différents fournisseurs d'échanger et de collaborer. Notre travail propose un nouveau protocole de communication inter-Cloud. Le protocole englobe l'échange des informations de routage, les demandes des services, la réservation et la gestion des erreurs.

3.2 Description du protocole :

Tout protocole est construit d'un ensemble de structure et de règle d'opération. Les structures doivent être explicites comme la longueur, la taille des champs et leurs ordonnancements. Le protocole doit prendre en considération l'environnement dans lequel il opère. Notre protocole proposé s'exécute dans plusieurs routeurs ou nœuds responsables de gestion de trafic à l'intérieur de Cloud. L'endroit exacte où le protocole s'exécute dans un Cloud est caché pour les communicants. Le système contient un ensemble de fournisseurs de service Cloud interconnectés via des liens à bande passante assurée (voir figure **Figure 30**). Les fournisseurs de service Cloud sont égaux (non hiérarchique, non centralisé) et donc peer-to-peer. Le protocole prend en charge le fait que le Cloud Computing est un système trop dynamique et que l'état de chacun doit être connu par les autres. Le protocole offre seulement le moyen logique pour effectuer des propagations de l'information.

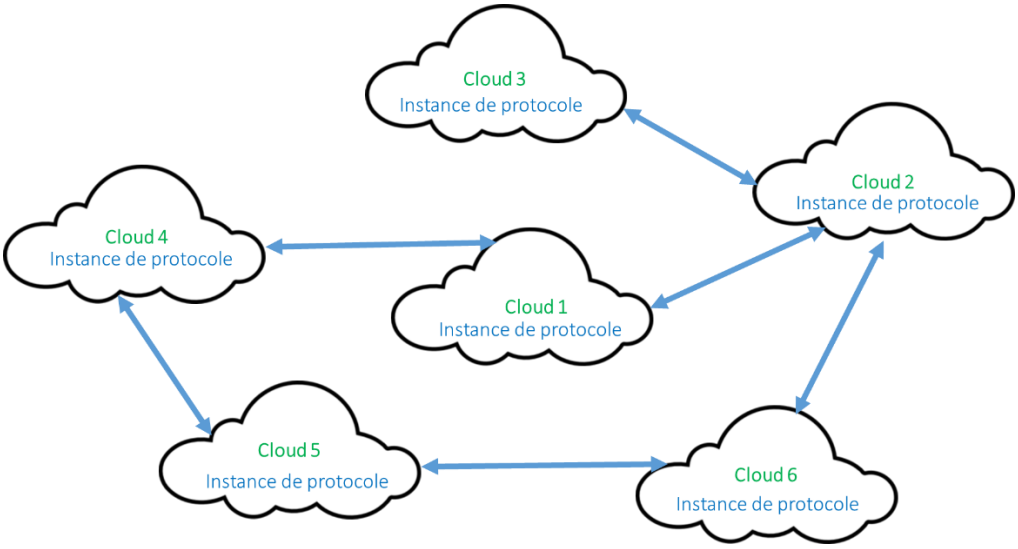

Figure 30: Représentation du Système d'environnement Cloud d'exécution de protocole proposé

Le protocole est conçu pour accepter plusieurs services offerts par le Cloud. il ne s'adresse pas seulement aux services standards comme le stockage. Le protocole s'ouvre sur tous types de service dans n'importe quel domaine. Nous présentons le protocole proposé dans deux parties. La première partie parle des structures utilisées, et la deuxième partie parle du fonctionnement, du traitement et la génération des messages de contrôle.

3.2.1 Structure des messages de protocole :

Notre protocole de communication décrit six structures de message de contrôle. On trouve le message UPDATE, REQUEST, REQUEST CONFIRMATION, RESERVATION ACCEPTAMCE, RESERVATION CONFIRMATION et NOTIFICATION (Erreurs). L'ensemble des messages sont encapsulés dans une entête BGP.

3.2.1.1 Entête :

L'entête utilisée dans l'encapsulation des paquets de notre protocole est la même que celle du BGP. Cette stratégie permet de différencier les paquets BGP de notre protocole, et en même temps bénéficier du mécanisme d'ouverture de session de communication et aussi la fiabilité de protocole de communication TCP. Nous rappelons que l'entête BGP contient trois champs ; Marker, Longueur (Length) et Type (Figure Header BGP). Les champs sont utilisés dans le nouveau protocole de la même manière qu'avec le protocole BGP.

3.2.1.2 Message UPDATE :

Le message UPDATE est le premier message échangé entre deux nœuds après une connexion TCP et puis une session BGP. Ce message contient une liste d'information sur les fournisseurs et les services qu'ils fournissent. Ces données incorporent aussi des données sur les chemins empruntés. A l'arrivée de la destination, le chemin complet sera enregistré dans le champ AS_PATH.

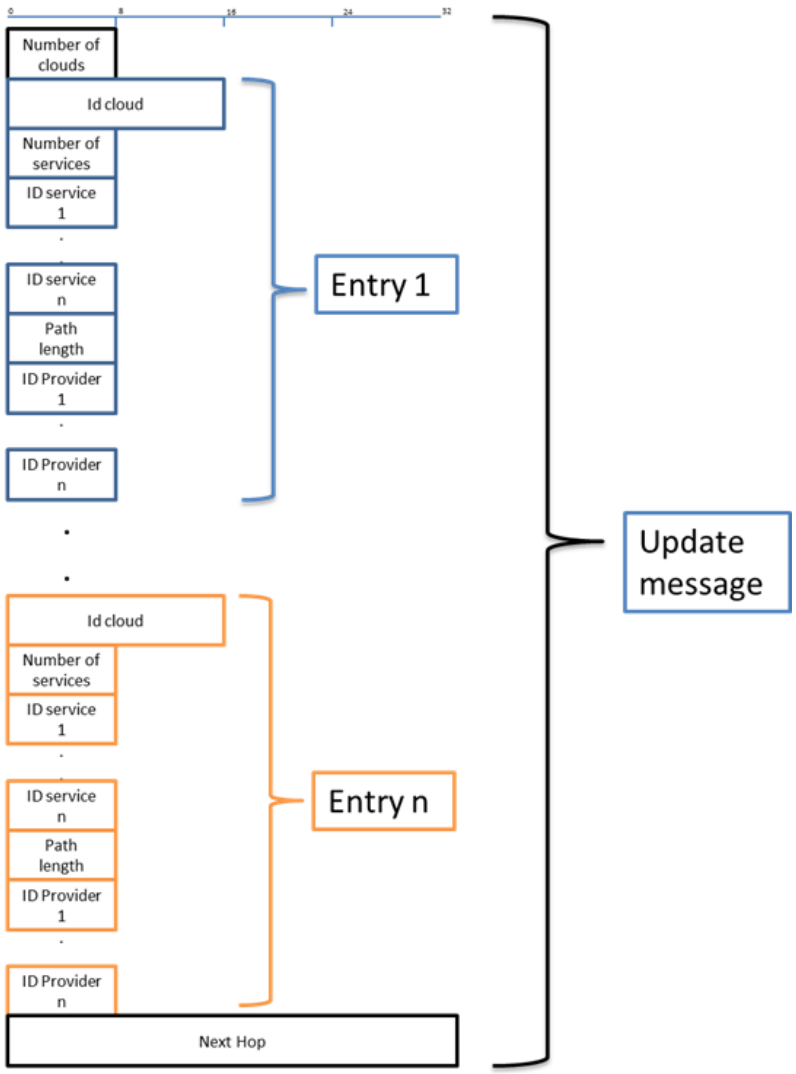

Figure 31: Structure de message Update

Figure 31décrit clairement la structure de message UPATE. Le message contient plusieurs listes. De droit à gauche. Le message UPDATE apparait comme un sous ensemble de sous structures répétées (Liste). Les sous-structures sont délimitées de deux champs. En haut le champ :

- ***Number of Cloud (Nombre de Cloud)***: Ce champ contient le nombre de Clouds inclue dans le message. Un id Cloud peut apparaitre plusieurs fois dans le même message UPDATE. La taille du champ est d'un Byte. Le message peut contenir 256 entrées. Avec condition que la taille globale ne doit pas dépasser le MTU (Maximum Transmission Unit).

En bas, le champ délimiteur est:

- ***Next Hope (Saut suivant):*** Ce champ de taille égale à 4 Bytes contient l'adresse IPv4 du nœud voisin. L'adresse IP et l'interface d'entrée vont déterminer la sortie des messages requête et autre. En d'autre terme ces deux informations seront utilisées dans le routage des services.

La grande accolade avec l'étiquette Update message qui montre la structure de message d'UPDATE et qui est composée des deux champs cités et de multiples structures étiquetées comme entry 1 … Entry 2 … Entry N ….

Chaque sous-structure (Entry) fait référence à un Cloud. Chaque sous structure est composée de :

- ***ID Cloud*** : Ce champ est de taille égale à 2 Bytes il contient l'identifiant du Fournisseur Cloud qui offre certains services.

Le champ ID Cloud est suivit d'une liste de service qui sont offerts par le Cloud mentionné dans le champ ID Cloud. Pour déterminer le nombre de service annoncé par le Cloud :

- ***Number of Services (Nombre de services)*** : ce champ est de taille égale à 1 Byte. De ce fait, chaque fournisseurs peut annoncer jusqu'aux 256 services à la fois.

La liste des services est sous forme de plusieurs Bytes contiguës. Chaque Byte représente un service :

- ***ID Service*** : ce champs de taille égale à 1 Byte contient l'identifiant d'un service offert par un Cloud, identifié dans le champ ID Cloud. Une succession de ce champ détermine l'ensemble des services offert par un fournisseur de service Cloud.

A la fin de la liste des champs ID Service, le champ PATH LENGTH apparait :

- ***PATH LENGTH (Longueur de lien)***: Ce champ a la taille égale à 1 Byte. Il contient la valeur de la longueur de champ suivant qui est une liste de plusieurs identifiants de fournisseurs Cloud en séquence construisant le chemin vers la destination.

Le dernier champ a exploré est le champ ID Provider ou ID Cloud. Comme montré dans la **Figure 31**, le lien se composent de plusieurs fournisseurs Cloud, la liste des ID provider est donc le conteneur de tous les fournisseurs de service Cloud intermédiaires avec inclusion de Cloud générateur de message.

Le message Update donc indique dans son premier champ combien de Cloud seront annoncées dans le message. À la fin de ce message c'est le champ qui contient l'adresse IP du nœud voisin qui sera dans le processus de routage. Au milieu de message UPDATE on trouve des structures de liste. La première liste de niveau 1 indique les fournisseurs. La deuxième liste de niveau 2 contient les services offerts par chaque fournisseur Cloud. La troisième liste de niveau 3 contient la séquence des identifiants des fournisseurs Clouds intermédiaires construisant le lien vers la destination. Ce message à la réception sera ingéré dans un processus qu'on le décrira dans la prochaine partie.

Après l'échange des messages UPDATE par les fournisseurs de service Cloud, les tables de routages convergent et incluront des informations suffisantes pour trouver un service et sa destination en cas de besoin. Les clients finaux ou les Clouds voulant demander un service

peuvent sélectionner de la table de routage un fournisseur et le communiquer. Le message REQUEST fait l'objet de cette action.

3.2.1.3 Message REQUEST :

Une des taches objectives de notre protocole consiste en la capacité d'échanger avec d'autres fournisseurs Cloud d'une façon autonome en cas de besoin. Notre vision logique considère qu'un Cloud devra contacter un autre Cloud automatiquement (Evènement programmée) pour lui demander un service. Dans ce cas, la structure de message était conçue pour répondre à ce besoin. La structure de ce message illustré dans la **Figure 32**:

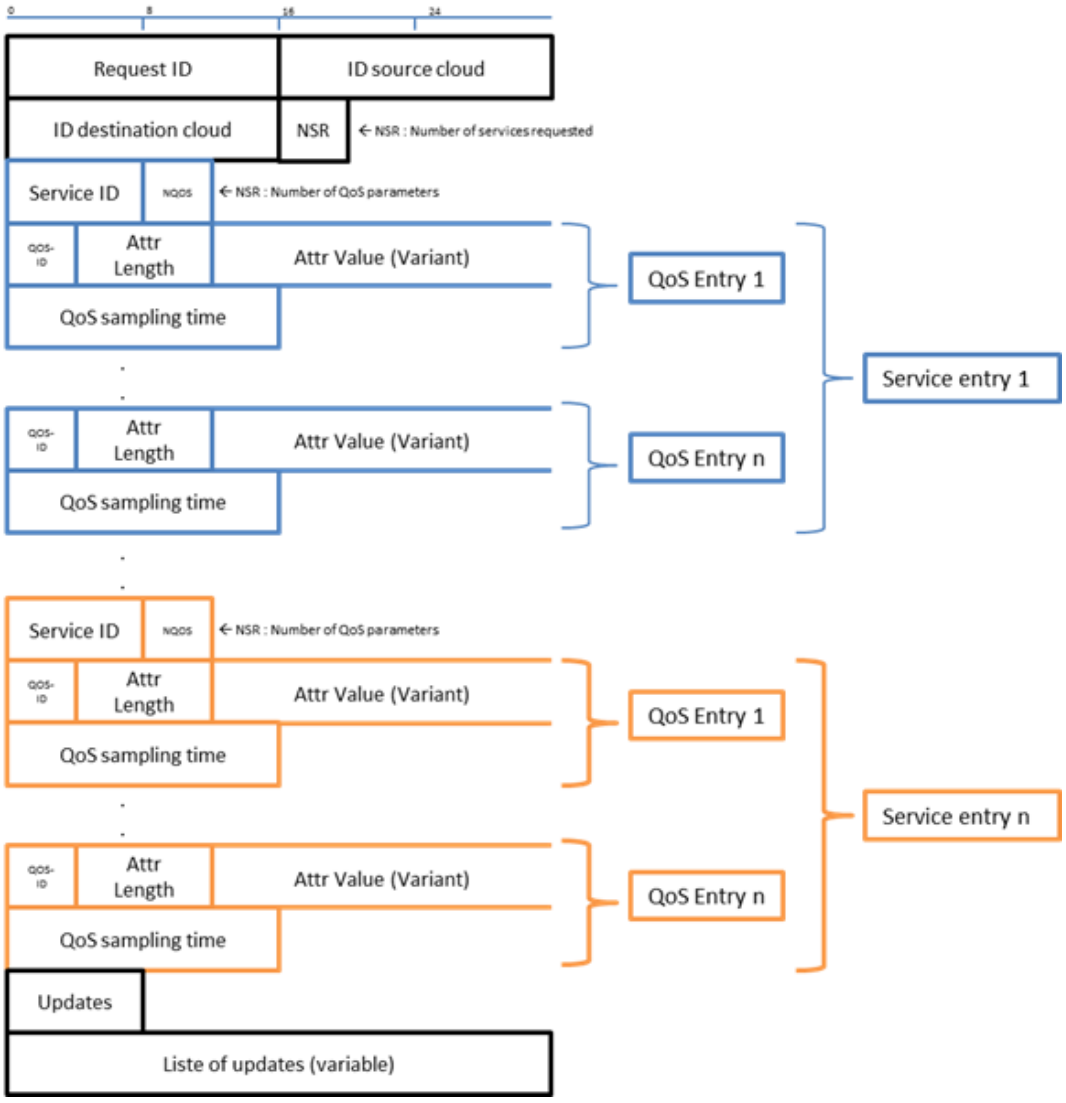

Figure 32 : Structure de message REQUEST

Le message REQUEST est le message le plus complexe parmi les structures des messages définies dans notre protocole. Le message REQUEST débute avec 4 champs communs pour l'ensemble des sous-structures comme le voit dans la **Figure 32.** Les 4 champs sont :

- ***Request ID (ID de requête) :*** Ce champ de taille égale à 2 Bytes contient un identifiant de la requête. L'identifiant est interne par rapport au fournisseur de

service Cloud générateur de la requête. Ce champ est rempli par le Cloud initiateur.

- ***ID Source Cloud*** : Ce champ de taille égale à 2 Byte contient l'identifiant de Cloud initiateur de requête.
- ***ID Destination Cloud :*** Ce champ de taille égale à 2 Byte contient l'identifiant de Cloud destinataire de requête.

Si le champ Request ID identifie la requête intérieurement par rapport au Cloud initiateur, le triplet des identifiants (Request Id, ID Source Cloud, ID Destination Cloud) différencie les requêtes par rapport à l'ensemble des fournisseurs Clouds interconnectés.

Comme un client peut demander plusieurs services à la fois, donc le message Request doit indiquer ce nombre. C'est le rôle de champ qui suit les trois champs précédents appelé NSR (Number of Services Requested).

- ***Number of Service Requested:*** Ce champ est limité à seulement 4 bits. C'est à dire qu'une requête de service peut demander seulement 16 services par message et vers une seule destination.

Dépendamment du champ Number of Service Requested, une liste de sous structure. Alors chaque structure déclare un service demandé en l'associant à une liste de paramètre de qualité de services exigés. Les champs se présentent comme suit :

- ***Service ID :*** ce champ de taille égale 1 Byte contient le code de service à demander.

Pour déterminer le nombre de paramètres de qualité de service associé à chaque service, un champ s'affiche dans chaque entrée (QoS Entry 1). Le champ nommé NQS (Number of QoS) est défini pour ce rôle.

- ***Number of QoS :*** Ce champ de taille égale à 4 bits contient le nombre des paramètres de qualité de service concernant le service demandé.

Une fois le nombre de paramètres de qualité de service sont connus, la liste des champs qui détermine les paramètres et les valeurs apparaissent toute en suivant le pattern (Type, Longueur, Valeur). Chaque entré contient les trois champs suivants :

- ***QoS-ID*** : ce champ de taille égale à 4 bits contient l'identifiant de type de paramètre de qualité de service en relation avec le service demandé.
- ***Length:*** Ce champ indique la longueur de champ suivant (Value) qui contient la valeur de paramètre exigé. Ce champ est de taille égale à 4 bits.
- ***Attr Value (Valeur d'attribut):*** ce champ a une taille variable. La longueur de ce champ dépend de type de paramètre. La taille de ce champ est déterminée par le champ précédent (Length).

A la fin de chaque entrée (QoS entry), un champ de 2 Bytes est collé pour indiquer le temps périodique d'envoi d'état de ce paramètre de QoS dans la session. Le champ est appelé :

- ***Qo Ssampling time (Temps d'échantillonnage de QoS):*** ce champ de taille égale à 2 Bytes. Il contient la fréquence d'échantillonnage de la valeur de QoS.

Arrivant à ce point, l'entrée QoS est définie, le reste du message requête est une succession de la structure. À la fin du message deux champs supplémentaires et finaux sont ajoutés.

Les champs sont :

- ***Updates*** : ce champ de taille égale à 1 Byte indique le nombre de fois oùlemessage a été mis à jour par les nœuds intermédiaires. Le champ est suivi par une liste d'information de mise à jour.
- ***Liste of updates*** : comme son nom l'indique, ce champ contient la liste des structures qui contient les informations du changement du paquet durant son transfert depuis la source jusqu'à la destination.

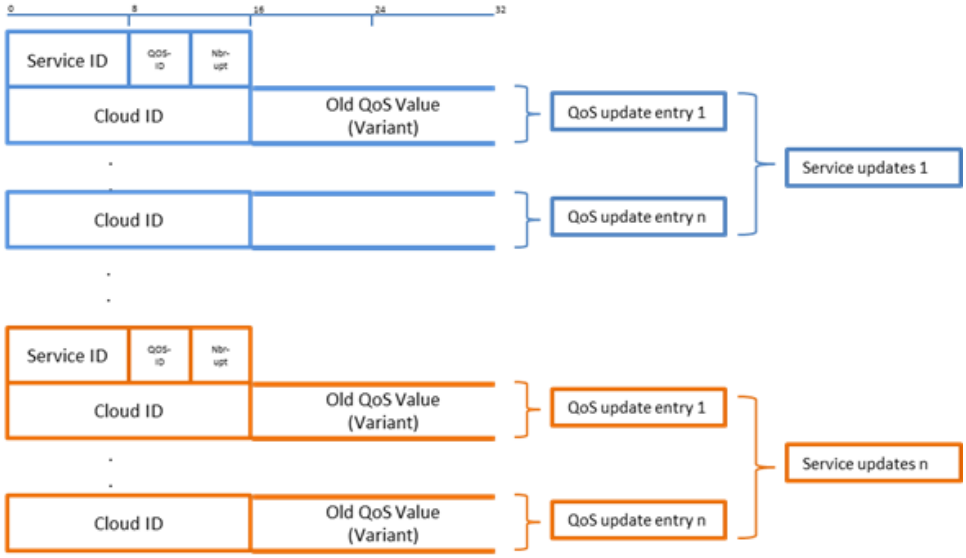

Figure 33 : Structure de champ List of Updates de message Request

Figure 33décrit comment les changements des valeurs des paramètres de qualité de service dans le message REQUEST sont structurés. Comme on peut voir, le champ est composé de sous structures et que chaque structure s'intéresse aux changements particulières de chaque service. La structure de niveau 1 combine les champs suivant :

- ***Service ID :*** indique le service que ses paramètres QoS ont été modifié par des nœuds intermédiaires. Sa taille est égale à 1 Byte.
- ***QoS-ID :*** ce champ de taille de 4 bits contient le code ou identifiant de paramètre de qualité de service appartenant au service indiqué dans le champ Service ID.
- ***Nbr-changes (Nombre de changement):*** ce champ de 4 bits détermine combien de fois le paramètre a été modifié. Les nouvelles valeurs seront insérées dans le champ principal de message Request par contre les valeurs anciennes sont empilées dans la liste des mises à jours.

Après le champ Nbr-changes, le nombre des misesà jour sera connu. Les champs suivant sont une liste de structures. Chaque structure contient les champs suivants :

- ***Cloud ID :*** ce champ de taille égale à 2 Bytes contient l'identifiant de Cloud intermédiaire qui a modifié et empilé l'ancienne valeur dans le champ de liste des updates.
- ***Old QoS Value :*** finalement ce champ contient la dernière ancienne valeur avant qu'elle soit modifiée par le fournisseur de service Cloud intermédiaire. De cette façon, tous les nœuds d'uplink seront au courant des modifications subies par le message Request et peuvent connaitre la valeur originale. Ces connaissances peuvent être utiles dans le calcul des ressources à mettre en disposition pour un service.

Le message REQUEST est initialisé par un fournisseur Cloud ou un client final vers un autre Cloud dont l'identifiant est connu. Le choix suit des critères déterminés par le fournisseur de Cloud initiateur. Les nœuds intermédiaires vont recevoir ce message, et appliquer les règles et politiques sur le paquet en suite il sera rediriger vers le nœud prochain. Le nœud modifie les paramètres et met à jours le champ update. Le paquet se propage de point à point jusqu'au nœud destinataire. Si la QoS était acceptée donc le nœud va générer un message ACCEPTANCE. Nous verrons dans ce qui suit les détails de message ACCEPTANCE.

3.2.1.4 Message ACCEPTANCE :

Un troisième message de protocole était défini, qui est le message ACCEPTANCE. Ce message supporte deux sous types de message dont PARTIAL ACCEPTANCE et FULL ACCEPTANCE. Les noms des deux messages sont clairs. Le premier message (PARTIAL ACCEPTANCE) est un message généré par le nœud destinataire quand des paramètres sont acceptés sauf que d'autre sont liée au nœud intermédiaire dans le sens downstream. Nous aurons plus de détails sur le concept de génération de message d'acceptation dans la deuxième partie des opérations et traitement des messages de contrôle de protocole. On est satisfait juste avec la structure de ce message à ce point-là. Le message ACCETANCE fait partie de la catégorie des messages de notifications.

3.2.1.4.1 MESSAGE FULL ACCEPTANCE

Le message ACCEPTANCE FULL généré seulement par le nœud destinataire et commuté par les nœuds intermédiaires du lien créé par le message REQUEST. Le message FULL ACCEPTANCE est un message très simple dans sa structure. Il est composé de 5 champs de taille fixe. Sa taille globale est égale à 8 Bytes (26 Bytes incluant l'entête (19 Bytes)).

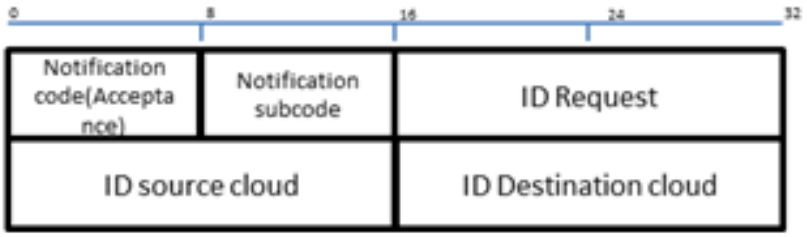

Figure 34 : Structure de message FULL ACCEPTANCE

Les 5 champs de message FULL ACCEPTANCE sont :

- ***Notification Code*** : le nom notification vient de message père NOTIFICATION. Un message ACCEPTANCE n'est rien qu'un message de notification. La taille de ce champ est égale à 1 Byte.
- ***Notification subcode*** : ce message est utilisé pour distinguer entre un message FULL ACCEPTANCE et un message PARTIAL ACCEPTANCE. sa taille est égale à 1 Byte.
- ***ID Request :*** ce champ est identique au champ Request ID de message Request. Ce champ contient la valeur attribue au début de la session par le fournisseur de service Cloud initiateur. Sa taille est égale à 2 Bytes.
- ***ID Source Cloud*** : ce champ de taille égale à 2 Byte contient l'identifiant de fournisseur de service Cloud qui a initialisé la requête (le message Request).
- ***ID Destination Cloud*** : ce champ de taille égale à 2 Byte contient l'identifiant de fournisseur de service Cloud qui a reçu la requête (le message Request).

Il est important d'observer le maintien des valeurs des champs ID Request, ID Cloud Source et ID Cloud Destination. Grace à ces trois champs, les nœuds intermédiaires et finaux peuvent faire correspondre entre les messages d'ACCEPTANCE et le message REQUEST (Idem pour les autres messages). De plus, il ne faut pas basculer entre les deux champs ID Source Cloud et ID Destination Cloud. En outre il faut les garder de la même façon dont ils étaient présentés dans le message REQUEST.

3.2.1.4.2 Message PARTIAL ACCEPTANCE

Le deuxième type de message D'ACCEPTANCE est le message PARTIAL ACCEPTANCE. Ce message a d'autres champs que les champs de message FULL ACCEPTANCE. Deux champs supplémentaires sont ajoutés.

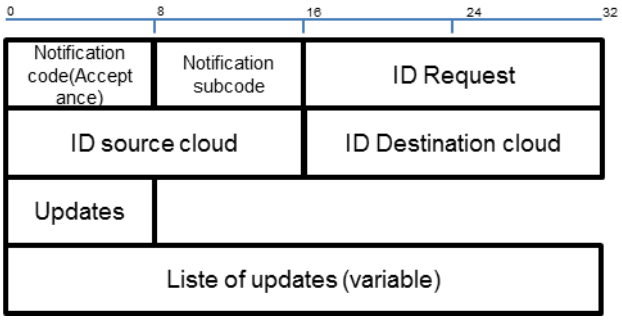

Figure 35 : Structure de message PARTIAL ACCEPTANCE

Le champ Updates et List of Updates jouent le même rôle que dans le message REQUEST. Certains services ont des paramètres de qualité de service qui doit être vérifiés dans le sens inverse (du destinataire à la source). Nous avons classifié les paramètres dans différents catégories. Cette classification nous a permis de bien contenir tous les cas possibles des paramètres de n'importe quel service. Les deux champs sont :

- ***Updates :*** il contient le nombre de mise à jour. Au moins il existe une mise à jour fait par l'initiateur de message PARTIAL ACCEPTANCE. ce champ est de taille égale à 1 Byte.
- ***List Of Updates :*** ce champ contient une liste des mises à jour des valeurs des paramètres de la qualité de service. Il suit le même processus des champs Update et list of updates de message REQUEST sauf que dans le cas de PARTIAL ACCEPTANCE, les nœuds peuvent seulement empiler des nouvelles valeurs car il n'existe pas de champs spécifiques qui contiendront les valeurs de QoS comme dans le message REQUEST.

Dans les deux cas, le message ACCEPTANCE est un message de contrôle de protocole qui fait partie d'une des phases de négociation des services automatiquement par le protocole entre les fournisseurs de service Cloud. À la réception de ce message, le nœud récepteur répond avec un message de confirmation de requête.

3.2.1.5 Message REQUEST CONFIRMATION :

Pour donner plus de fiabilité a notre protocole. Nous voyons que la requête et la réservation des ressources doivent être confirmées par les deux nœuds finaux. Dans un premier cas, la première confirmation est celle de Requête. Lorsque le nœud initiateur de requête reçoit un message d'acceptation, ce dernier enverra le message REQUEST CONFIRMATION. La structure de message est la même que la structure parentale « NOTIFICATION ». Les champs de message REQUEST CONFIRMATIONsont les suivants :

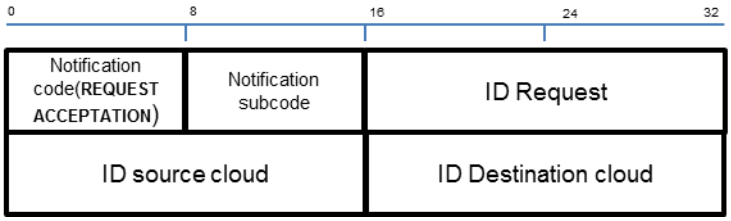

Figure 36 : Structure de message REQUEST CONFIRMATION

On remarque bien la ressemblance entre le message FULL ACCEPTANCE et le message REAUEST CONIRMATION. Le protocole différencie les deux messages via le code de notification. Le plus important est de garder les mêmes valeurs dans le champ ID Request, ID Source Cloud et ID Destination Cloud. Ces champs sont responsables du bon fonctionnement de protocole et la cohérence des données durant la session de négociation. Le message est envoyé vers le Cloud destinataire. Les messages intermédiaires font passer le message sans le modifier.

À la réception de message REQUEST CONFIRMATION par le nœud destinataire, ce dernier confirme la requête, et renvoie le dernier message dans la session de protocole pour la confirmation de réservation. Le message RESERVATION CONFIRMATION est responsable de cette action. Ce message nécessite l'accord de tous les nœuds intermédiaires. En recevant le message RESERVATION CONFIRMED, le nœud commence l'exploitation de service demandé.

3.2.1.6 Message RESERVATION CONFIRMATION

De la même manière que le message REQUEST CONFIRMATION, ce message est envoyé pour confirmer la réservation. La structure de ce message est la même qu'un simple message de notification mais avec une extension à la fin pour inclure les paramètres nécessaires pour exploiter le service. La structure de message RESERVATION CONFIRMATION est illustrée dans la **Figure 37.** Le message contient les champs suivants :

- ***Notification code*** : ce champ contient le code indiquant que le message de notification est un message de confirmation de réservation. La taille de ce champ est égale à 1 Byte.
- ***Notification subcode*** : Ce champ contient un code qui donne beaucoup de spécificité. Ce champ n'est pas défini actuellement pour le message de confirmation de réservation. La taille de ce champ est égale à 1 Byte.
- ***ID Request*** : contient l'identifiant de la requête donnée par le Cloud demandeur. ce champ à la taille de 2 Bytes.
- ***ID Source Cloud*** : Ce champ contient l'identifiant de Cloud initiateur de requête. La taille de champ est égale à 2 Bytes.

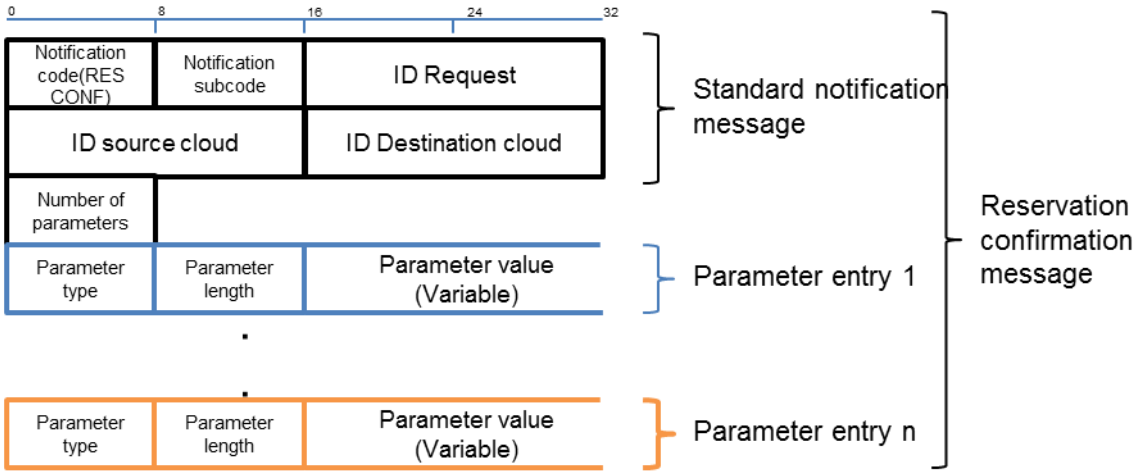

Figure 37 : Structure de message RESERVATION CONFIRMATION

- ***ID Destination Cloud*** : ce champ contient l'identifiant de fournisseur Cloud destinataire. La taille de ce champ est égale à 2 Bytes :

A partir de ces derniers champs. Le message RESEVATION CONFIRMATION ajoute deux champs supplémentaires. Le premier indique le nombre de paramètres existants dans la liste des paramètres.

- ***Number of Parameters*** : ce champ de taille égale à 1 Byte contient la valeur du nombre de paramètres inclus dans le message RESERVATION CONFIRMATION

Le champ suivant est une liste de paramètres dirigé par le pattern (Type, Longueur, Valeur). Chaque entrée de la liste paramètre est composée des trois champs suivants :

- ***Parameter Type***: Ce champ de taille égale à 1 Byte contient le type de paramètre lié au service demandé.
- ***ParameterLength*** : ce champ de taille égale à 1 Byte contient la longueur de la valeur de paramètre de service. Ce champ permet de connaitre la taille de champ suivant.
- ***Parameter Value*** : ce champ de taille variable contient la valeur de paramètre de service.

A la réception de message RESERVATION CONFIRMATION, le nœud peut commencer l'exploitation de service. Grace aux paramètres inclus dans le message de confirmation, le nœud demandeur peut configurer une connexion à ce service. Les fournisseurs de service Cloud qui composent le backbone du réseau peer-to-peer doivent reconnaitre au paravent les différents types de paramètres de chaque service et les codes des services et les contraintes de qualité de service.

Une session de protocole proposé est considéré complète lors de la réception de message RESERVATION CONFIRMATION. Ce service est gardé ouvert tant que le Cloud client est toujours en exploitation de service. Durant la session de protocole, des erreurs peuvent

apparaitre. Nous avons pensé à ce cas de gestion d'erreur en définissant un message de notification des erreurs.

3.2.1.7 Message ERROR NOTIFICATION :

Durant une session de négociation de service Cloud, il est envisageable que des erreurs peuvent reproduire. Dans ce contexte-là, le message ERROR NOTIFICATION informe tous les nœuds intermédiaires et finaux sur l'erreur en envoyant ce message. Le message indique le type d'information ou d'erreur. Pour qu'il soit plus explicite il utilise deux champs supplémentaires, le champ subcode et le champ ErrorDetails.

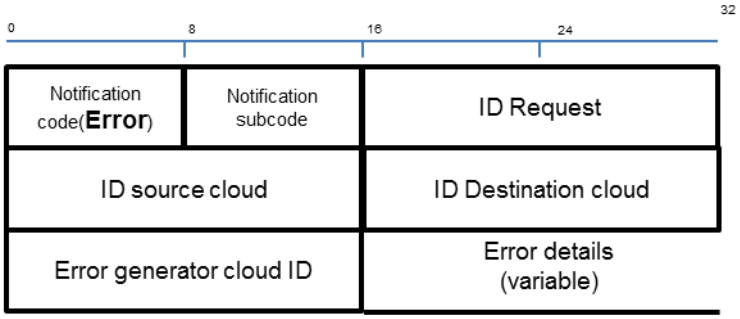

Figure 38 : Structure de message ERROR NOTIFICATION

Le message ERROR NOTIFIICATION peut être généré par n'importe quel nœud qui forme le chemin entre le Cloud source et le Cloud destinataire. Ce message est un sous message de message notification. Il contient le même champ qu'un message de notification simple. Le message contient les champs suivants :

- ***Notification code (Error) :*** ce champ contient le code d'erreur. Il est de taille égale à 1 Byte.
- ***Notification subcode*** : ce champ spécifie de plus le code d'erreur. Dans le cas où l'erreur est divisé en sous classes, le champ notification subcode peut identifier alors chaque classe d'erreur.
- ***ID Request*** : ce champ de taille égale à 2 Bytes contient l'identifiant de la requête attribué par le nœud demandeur de service.
- ***ID Source Cloud:*** Ce champ de taille égale à 2 Bytes. Il contient l'identifiant de Cloud Source.
- ***ID Destination Cloud :*** Ce champ de taille égale à 2 Bytes. Il contient l'identifiant de Cloud Destinataire.

Deux champs supplémentaires sont ajoutés à la structure standard de message de notification. Ces champs sont :

- ***ErrorGenerator Cloud ID:*** Ce champ contient l'identifiant de nœud Cloud qui génère le message d'erreur. Le message d'erreur peut être envoyé dans le sens

upstream, downstream ou les deux en même temps. La taille de ce champ est égale à 2 Bytes.

- ***ErrorDetails:*** Ce champ de taille variable contient des détails sur l'erreur. Les données peuvent suivre un format bien défini (XML) ou sous forme de simple texte ou des données binaires.

Lors de la réception de ce message, la session de négociation de service Cloud sera fermée et les ressources sont libérées.

3.2.1.8 Table de Routage Des Services:

La table de routage standard de protocole de négociation de service se compose de quatres colonnes principales. Le protocole prend en considération la possibilité d'extension de protocole BGP car les métriques des services changent en fonction du service offert. Il peut exister plusieurs types de services offerts de différents domaines. Chaque domaine a ses spécificités. Nous avons choisi de garder le protocole flexible à des extensions possibles pour pouvoir inclure tous les types de service possible. La **Figure 39**décrit la structure globale qui contient les colonnes suivantes :

- ***ID CLOUD*** : Cette Colonne contient la liste les fournisseurs de services Cloud qui ont annoncés leurs services via le message UPDATE.
- ***ID SERVICE*** : cette colonne contient la liste des services offert par les Clouds. Chaque Service de la colonne ID SERVICE correspondant à gauche a son fournisseur Cloud.
- ***AS PATH*** : Cette Colonne et pour chaque service fourni par un fournisseur Cloud contient une liste d'identifiants. L'ensemble des identifiants sont la liste des Clouds qui forment et qui mènent vers le Cloud en question.
- ***Exit Network Address***: cette colonne affiche les interfaces de sortie (Routeur) pour atteindre le réseau de Cloud qui offre le service.
- ***Extension*** : cette colonne est une représentation symbolique de la possibilité du protocole d'ajouter d'autres colonnes à la table de routage qui sont déterminé par chaque fournisseur Cloud.

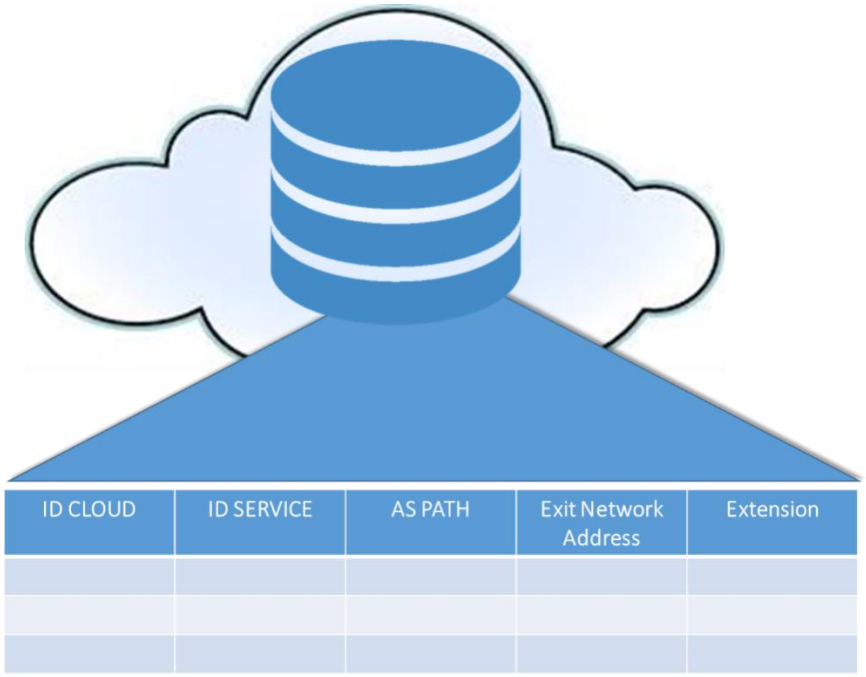

Figure 39 : Structure Standard de Table de routage des services

L'une des clefs de la flexibilité du protocole proposé est la possibilité d'étendre la table de routage pour que tous les types de services puissent utiliser les métriques qui leur sont spécifiques à lui. La métrique de base pour le routage des services dans notre protocole est le nombre de saut qui est défini par le champ AS PATH. On dit que l'extension de la table de routage service est Cloud-driver. Dans un sens que c'est au fournisseur Cloud de déterminer l'architecture extensible de la table de routage des services. Avec l'existence de l'adresse de sortie (Exit network Address), notre protocole peut créer l'interface dans sa table de routage et la table de routage réseau crée par des protocole de routage standard comme RIP, OSPF, ISIS ou autres.

3.2.2 Les Operations de Protocole

Une partie du protocole était consacrée aux structures des objets qui forment les messages et la table de routage de notre protocole était présentée dans la première partie. Les messages de contrôle de session de négociation de service Cloud nécessitent une définition de la façon dont ils doivent être traités. Nous détaillons le traitement de chaque message et on finira par une présentation qui englobe l'ensemble des échanges entre les nœuds durant une session.

3.2.2.1 Message UPDATE :

Après l'ouverture d'une session BGP, chaque nœud (Cloud) échange avec ses voisins un message UPDATE. Chaque nœud va contenir l'ensemble des informations de routage des services. Les informations de routage de service sont tirées depuis la table de routage de service. Pour minimiser la taille des données échangées, les services qui sont atteignables à travers le même chemin sont regroupés. Une liste de fournisseurs de service Cloud est créée et chaque entrée de la liste contient une liste des services qui peuvent être atteints via le même chemin. Il devient possible qu'un fournisseur apparaisse plusieurs fois dans une liste

de message UPDATE. Un fournisseur Cloud peut apparaitre plusieurs fois dans le cas où il est atteignable via plusieurs chemins. Le nœud responsable de la propagation des messages UPDATE ne doit pas retransmettre les informations à travers la même interface dont ils ont été reçus. Le message UPDATE est transmis périodiquement mais seulement dans le cas de changement dans le contexte de routage des services ou bien un évènement explicite qui demande le renvoie des informations du routage de nouveau. Les données dans la table de routage de service ne sont pas contiguës dans la mémoire. Pour cette raison, avant l'envoi des messages ; un processus de formatage est exécuté en transformant des structures non contigües en message contiguë pour le transmettre au nœud suivant **Figure 40**.

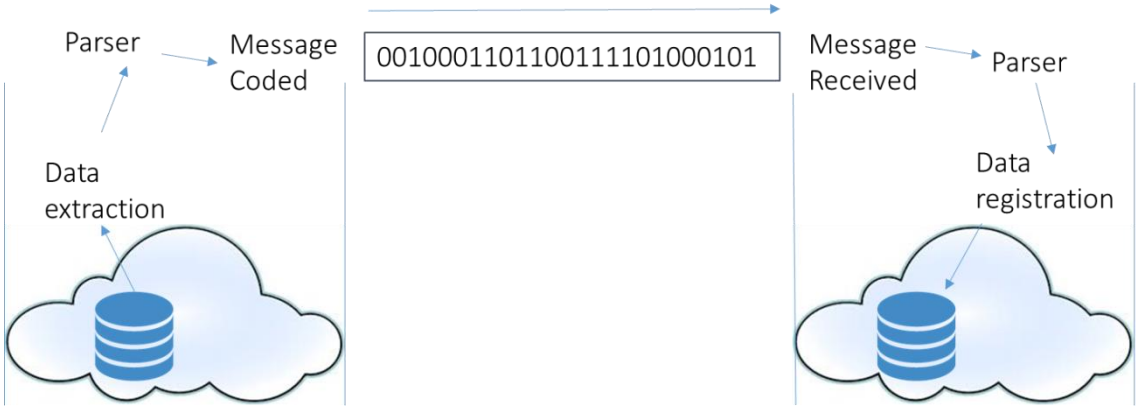

Figure 40 : Generation de message UPDATE

Lors de réception de message UPDATE, il passe par un processus de traitement. Ce processus est illustré dans la **Figure 41.** Lors de la réception d'un message UPDATE, ce dernier doit passer par le module de sécurité. Les règles de sécurités diffèrent chaque fournisseur de service Cloud. Les règles peuvent être appliquées à différents niveaux. Par exemple dans le niveau de la couche transport de modèle ISO, il peut être appliqué au niveau de l'entête de protocole, il peut être appliqué sur les identifiants source/destination, il peut être appliqué sur le type de service …etc.

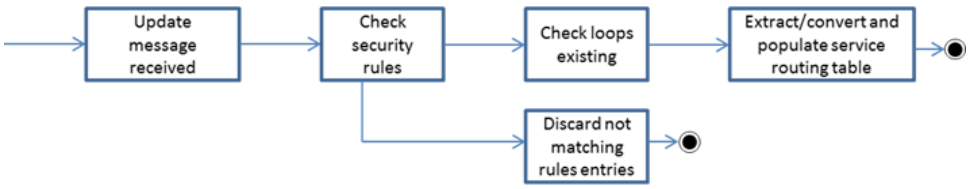

Figure 41 : Traitement de message UPDATE a la réception

Dans le cas où le message ne passe pas les règles et politiques de sécurité, le service ou le Cloud en question sera écarté. Dans le cas contraire, le message sera transformé en ligne d'entrée qui contient des informations qui seront transformées. Les entrées acceptées seront insérées dans la table de routage. Les nouvelles entrées seront propagées dans le réseau pour informer les autres Clouds sur les nouvelles routes obtenues. La retransmission est permise dans le cas ou :

- Des nouvelles informations sont disponibles

- Le timer a expiré

Après la convergence des tables de routage, les fournisseurs de service Cloud seront prêts à recevoir des requêtes de service et négocier des services distants.

3.2.2.2 Message REQUEST :

Le message REQUEST permet à un fournisseur de demander un service auprès d'un autre fournisseur Cloud. La requête est initialisée manuellement ou automatiquement par un évènement programmé. En cas de nécessité de ressource ou d'un service, un fournisseur crée un message REQUEST. Le Cloud choisira dans la table de routage un Cloud qui fournit le service en besoin. Le processus de sélection de service est Cloud-driver, c'est-à-dire que la méthode et l'algorithme de sélection des chemins est défini par le fournisseur de service Cloud. Un fournisseur peut demander plusieurs services au même fournisseur de service Cloud. Donc le Cloud sélectionne un fournisseur et les services demandée. Il associe à chaque service une liste de paramètres. Comme il peut aussi ne pas spécifier ces paramètres. Le deuxième point qui permet à notre protocole d'être adaptif et générique c'est le fait de permettre aux fournisseurs de service Cloud de définir leur propre façon de sélectionner les entrées en se basant sur des métriques spécifiques au service et au domaine. La conception de protocole a toujours pris la contrainte de l'existence de multitude de services et l'hétérogénéité des technologies, méthodes, algorithmes, plateforme, service, etc. Cette caractéristique lui donne l'avantage pour qu'il soit intégré sans atteindre les principes de fonctionnement interne des infrastructures des fournisseurs de service Cloud.

Lors de la réception de message REQUEST, le message subira plusieurs vérifications. L'implémentation de ce processus peut changer dans l'ordonnancement des conditions dans le but d'améliorer et de diminuer le nombre des opérations par message.

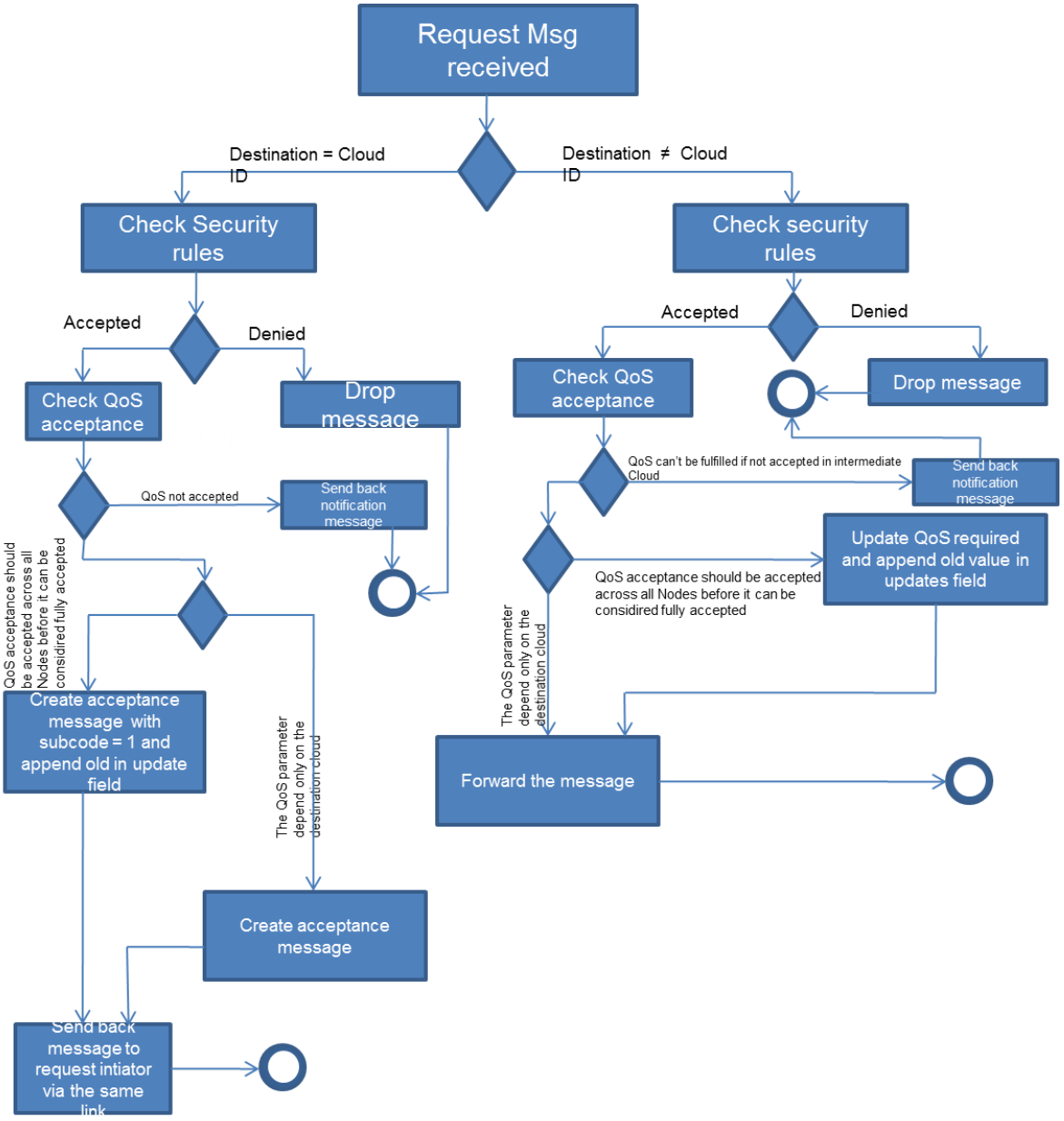

Figure 42 : Processus de traitement de message REQUEST a la réception

Le schéma présenté dans la **Figure 42**résume l'ensemble des opérations de traitement de message REQUEST lors de la réception par un nœud destinataire ou intermédiaire. En haut, le message est reçu par un nœud de fournisseur de service Cloud, ce dernier vérifie si le message est destiné à lui-même ou un autre nœud Cloud. La vérification se fait en se basant sur le champ destination. Si l'identifiant dans le champ ID Cloud destination n'est pas égale à celui de l'identifiant de récepteur de message, passe à la vérification des règles de sécurité implantées par le fournisseur de service Cloud lui-même. Après la vérification, soit le message sera accepté soit il ne sera pas et dans ce cas le message sera totalement ignoré. Si le message est accepté après la vérification des règles de sécurité, le message passe une autre étape qui est celle de la vérification d'acceptation des paramètres de qualité de service. Si le message contient un paramètre de QoS qui est lié aux nœuds intermédiaires et que ce paramètre doit être accepté par tous les nœuds et que le nœud récepteur intermédiaire ne peut

pas répondre à la valeur demandé par le Cloud initiateur alors ce dernier arrêtera le processus de négociation de service et envoie un message de NOTIFICATION au nœud initiateur. Si les paramètres de QoS correspondants au service nécessitent un cumul de tous les nœuds intermédiaires et finaux, le nœud intermédiaire ne peut pas décider si le paramètre et totalement accepté ou pas alors dans ce cas, le nœud va mettre à jour la nouvelle valeur de qualité de service demandée et ajoute l'ancienne valeur dans la fin de message dans le champ List of updates et il incrémente la valeur de champ updates. Après ça, le nœud intermédiaire passe le message mis à jour au nœud suivant tout en faisant référence à la table de routage des services local. Alors, si les paramètres n'ont rien à voir avec les nœuds intermédiaire, le nœud ne sera qu'un transitaire et fait passer le message REQUEST au nœud suivant en faisant référence à la table de routage des services locaux.

Revenant au point du début, dans le cas où l'identifiant de nœud qui reçoit le message est égale à l'identifiant contenu dans le champ ID Service Destinataire. Le message passera toujours par le module de sécurité. Si le message n'est pas accepté pour des raisons de sécurité et politique de traitement, le message sera simplement ignoré. Dans le cas contraire, si le message est accepté pour qu'il soit traité. Le message passera à la deuxième étape qui vérifie l'acceptation des paramètres de qualité de service demandé. Si le nœud destinataire n'accepte pas la demande pour des raisons de disponibilité des ressources qui peuvent répondre aux valeurs des paramètres de qualité de service existant dans le message REQUEST, le nœud destinataire enverra un message NOTIFICATION au nœud initiateur de requête et la session sera fermée. Si le nœud accepte la requête, il aura deux cas. Le premier cas, c'est lorsque les paramètres de qualité de service de la requête nécessite une confirmation de tous les nœuds intermédiaires et la réponse finale ne se limite pas juste par rapport au nœud destinataire. Alors, à la réception du message par le nœud destinataire, ce dernier va créer un message d'acceptation (PARTIAL ACCEPTANCE) et il ajoute la nouvelle valeur de chaque paramètre de Qualité de service qui nécessite une acceptation de tous les nœuds intermédiaires dans le sens downlink. Dans le deuxième cas, les paramètres de qualité de service de la requête sont seulement liés au nœud destinataire, alors ce dernier va définitivement confirmer l'acceptation totale de la demande de service avec les paramètres de qualité de service exigés et dans ce cas contrairement au premier, le nœud génèrera un message d'acceptation FULL ACCEPTANCE au lieu de PARIAL ACCEPTANCE et il l'enverra dans le sens inverse vers le nœud initiateur de la requête.

La génération du message ACCEPTANCE (FULL ou PARTIAL) n'indique pas forcement que les ressources sont réservées. Pour cela, les deux nœuds doivent confirmer la requête et la réservation. Donc les deux nœuds vont attendre une réponse de confirmation dans les deux sens après la réception de message ACCTENACE. Ceci donne plus de fiabilité au protocole et à la négociation. Nous verrons maintenant les opérations liées au message ACCEPTATION.

3.2.2.3 Message ACCEPTANCE :

Le message ACCEPTANCE est un message généré par le nœud destinataire qui doit répondre en dernier sur la requête de demande de service. Ce message se divise en deux types. Un message d'acceptation complète (FULL ACCEPTANCE) et un message d'acceptation partielle. La **Figure 43**montre l'ensemble des opérations que le message suit dans un nœud intermédiaire ou final (initiateur de requête). A la réception de message d'acceptation, on vérifie si le message est destiné au nœud récepteur ou pas en comparant l'identifiant dans le champ ID Cloud source. Si la valeur dans ce champ n'est pas égale au identifiant local de Cloud qui reçoit le message, en vérifiera encore si c'est un message FULL ACCEPTANCE ou PARTIAL ACCEPTANCE. Si le message est FULL ACCEPTANCE, le nœud ne fait que passer le message au nœud suivant dans le sens inverse (downstream : Cloud demandé vers le Cloud demandeur de service).

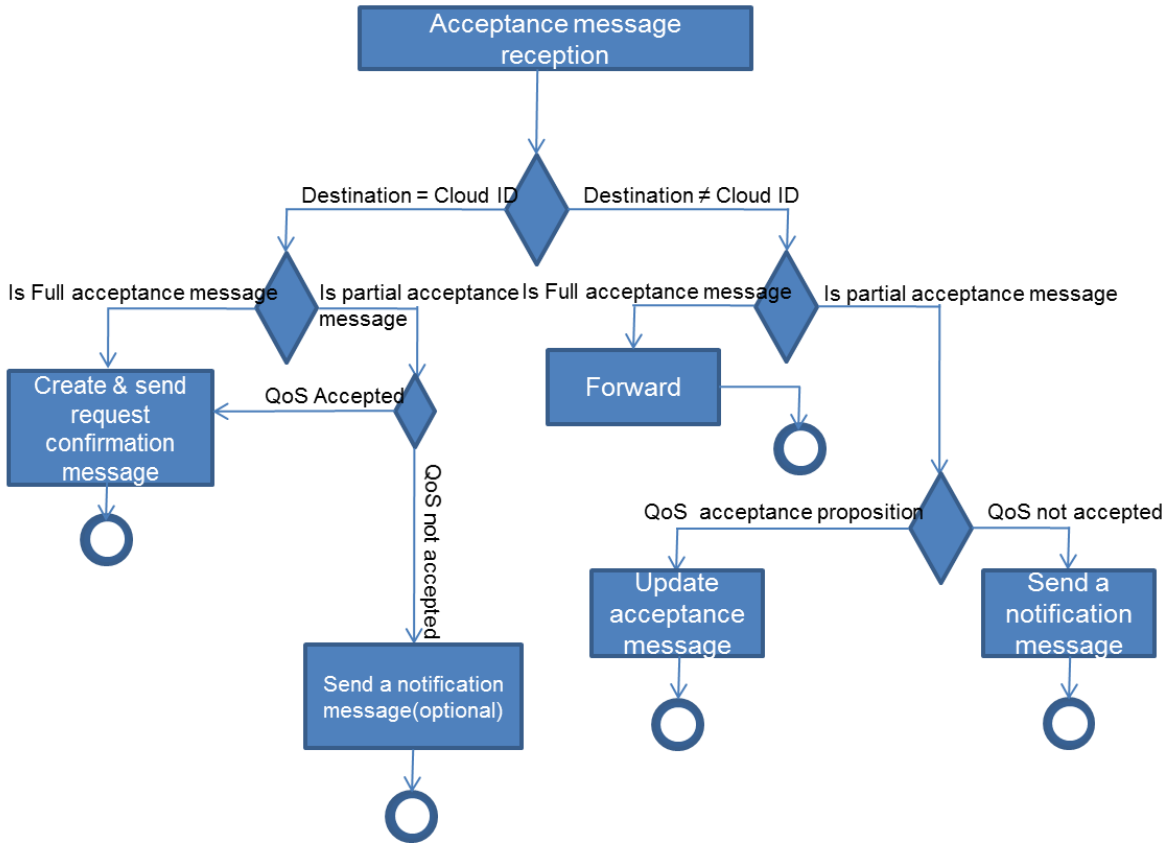

Figure 43 : `Processus de traitement de message ACCEPTANCE lors de la réception

Si le message est un PARIAL ACCEPTANCE, on vérifie l'acceptation des paramètres de QoS de service demandé. Si le paramètre dépend de tous les nœuds suivants et que ce dernier accepte une certaine QoS, alors il met une mise à jour de la valeur demandée du paramètre de qualité de service et il le fait passer au nœud suivant. Si le paramètre doit être accepté par tous les nœuds, la session sera fermé dès qu'un nœud final au intermédiaire refuse la demande. Dans ce cas, si ce nœud refuse la demande en raison de disponibilité des ressources, alors la session sera fermé et un message de notification sera envoyé dans les deux sens, c'est à dire vers le nœud demandeur qui a initialisé la requête et le nœud

générateur de message d'acceptation (nœud destinataire). Dans tous les cas, le message passera au system de sécurité et de politique définie par le propriétaire de Cloud. revenant au début du schéma illustré dans la **Figure 43**, dans le cas où la valeur de champ ID Cloud Source est égale à l'identifiant de Cloud local (récepteur). Dans ce cas, le message aatteint sa destination finale. Le dernier nœud dans le lien (initiateur de requête) vérifie encore les règles de sécurité interne, ensuite il continue avec la vérification de l'acceptation des paramètres de qualité de service. Il parait un peu ambigu si le nœud qui a demandé le service lui aussi vérifie ses capacités d'acceptation des paramètres de qualité de service. Nous expliquons celle par rapport aux paramètres qui nécessitent une accumulation de tous les nœuds intermédiaires. Alors, chaque nœud met à jour sa valeur et demande une nouvelle valeur qui doit être acceptée par tous les nœuds suivants. Pour cela, le nœud initiateur de requête devient lui-même une entité qui prend la décision sur l'acceptation de la requête. Si après cette succession de processus exécuté par les nœuds intermédiaires et final, les paramètres de qualité de service liés au service Cloud demandé sont acceptés, le nœud initiateur de requête va régénérer un nouveau message qui est le message REQUEST CONFIRMED.

3.2.2.4 Message REQUEST CONFIRMATION :

A la réception d'un message d'acceptation, le nœud initiateur de requête va générer un message REQUEST CONFIRMATION. Ce message a pour but de confirmer que le nœud initiateur a reçu l'acceptation de nœud destinataire et que les nœuds intermédiaires ont eux aussi accepter les paramètres de qualité de service. Ce message est envoyé vers le nœud suivant en chemin directe vers le nœud destinataire. Les nœuds intermédiaires recevant ce message vont le faire passer de nœud en nœud jusqu'au nœud destinataire. Les nœuds intermédiaires sont capables d'annuler la session durant cette phase de négociation de service. De plus, le message n'échappe pas aux règles de sécurité. Le message REQUEST CONFIRMATION est un message simple qui se base sur la structure de message NOTIFICATION. Lorsque le nœud final reçoit le message REQUEST COMFIRMATION, il continue son accord à fournir ce service, et il génère un message RESERVATION CONFIRMATION.

3.2.2.5 Message RESERVATION CONFIRMATION :

Lorsque le message REQUEST CONFIRMATION est reçu par le nœud destinataire, le nœud final comprend que la partie négociation est bien accepte par tous les nœuds et que le nœud initiateur a vraiment besoin du service. Dans ce cas, le quatrième et dernier message nécessaire dans le processus de négociation de service de protocole est génèré. C'est le message RESERVATION CONFIRMATION. Ce message contient aussi des paramètres à utiliser par le client pour qu'il puisse communiquer et utiliser le service contrairement au message REQUEST CONFIRMATON qui ne contient aucune information en plus que la structure simple d'un message NOTIFICATION. Ce message est passé d'un nœud à l'autre dans le sens inverse de chemin jusqu'au nœud initiateur de session. Ce dernier en recevant ce message peut commencer à exploiter le service en utilisant les paramètres fournis par le Cloud fournisseur. Durant et après l'ouverture de session de négociation automatique d'un

service Cloud avec le protocole proposé, des problèmes et des erreurs peuvent apparaitre. Ces derniers sont propagés grâce au message NOTIFICATION.

3.2.2.6 Message NOTIFICATION

Durant la négociation d'un service, il arrive parfois de rencontrer des problèmes et des erreurs de communication ou autres. Dans ce cas un message de notification est envoyé aux nœuds en question. Tout dépend de la phase de session, le nœud décide d'envoyer un message de notification en indiquant le type d'erreur ou d'information inclues dans le message. Le message peut être envoyé au deux nœuds finaux (source et destinataire) ou bien à un seul. Tous les nœuds ont cette capacité d'initialiser un message d'erreur (NOTIFICATION) qu'ils soient des nœuds finaux ou des nœuds intermédiaires. En cas d'erreur fatal, la session sera fermée et les ressources seront libérées. Le nœud par la suite peut relancer le processus de nouveau suivant ses politiques.

3.2.2.7 Processus de sélection de route d'un service

Le point fort du protocole est sa capacité d'inclure autant de services que les Clouds peuvent offrir. Grace à l'indépendance de la structure de table de routage et le processus de sélection de route vers les Clouds de destination du protocole, le Cloud peut intégrer leurs services spécifiques à son domaine et qui détermine la stratégie de choix d'un fournisseur par rapport un autre. Donc le processus de sélection et la structure de table de routage des services extensibles dépendent du fournisseur de service Cloud et pas du protocole. En dépit de la capacité des fournisseurs de service Cloud d'étendre la table de routage de service, ces derniers doivent garder la structure de base qui se compose de champ ID /Cloud, ID Service AS_PATH et Exit interface. Le processus simple de sélection de route se base sur le chemin le plus court. Les fournisseurs du service Cloud peuvent élaborer un modèle plus avancé pour la sélection de fournisseur en se basant sur des méthodes stochastique et probabiliste, sur des informations requisent par expérience, des données préconfigurées. Le processus de sélection du meilleur fournisseur peut faire l'objet d'un travail de recherche.

3.3 Classification des paramètres de qualité de service

Les paramètres de qualité de service diffèrent d'un service à un autre. Avec l'apparition de la technologie Cloud, les services ont été migrés vers des infrastructures externes. Les clients peuvent implémenter des services dont la performance de chaque service dépend d'un ensemble de métriques spécifiques au domaine du service. Un protocole de qualité de service doit prendre en considération tous ces paramètres spécifiques au service. Il n'est pas possible de connaitre tous les paramètres des services pour définir à chaque service une structure particulière. En revanche, le protocole doit prendre en considération tous les services offerts par la technologie Cloud. Un protocole de négociation de service avec exigence de qualité de service peuvt se limiter au service répandu dans le domaine de communication comme la bande passante, le débit, le stockage …etc. sauf que le Cloud ne se limite pas à ces services mais à plusieurs types de services. Pour permettre au protocole d'accepter tous les paramètres des différents services, nous avons identifié les classes possibles des paramètres de qualité de services. Les classes identifiées sont :

- Les paramètres liés uniquement au Cloud destinataire : ces paramètres seront juste vérifiés par le fournisseur destinataire. Si ce dernier accepte ces paramètres alors, il sera considéré comme acceptable dès qu'il génère le message d'acceptation.
- Les paramètres qui sont liés au nœud intermédiaire et final : ces paramètres nécessitent une acceptation par tous les nœuds. Ces paramètres peuvent être classés en deux sous catégories. Soit le paramètre est cumulatif ou non et si le paramètre est vérifié en front montant ou descendant.
- Paramètre cumulatif : l'acceptation du paramètre nécessite de voir est-ce que chaque nœud de lien peut donner comme ressource pour savoir à la fin si le paramètre peut être accepté.
- Paramètre obligatoire : ce type de paramètre doit avoir l'accord de tous les nœuds intermédiaires et si un seul des liens n'accepte pas le paramètre, la décision peut être prise pour dire que le service ne peut pas être servi via ce chemin.
- Paramètre a front montant (Upstream): dans cette sous classe la vérification de l'acceptation de paramètre se fait par tous les nœuds et la décision est prise par le nœud destinataire. La vérification se fait dans le sens Cloud initiateur au Cloud final (Destinataire).
- Paramètre à front descendant (Downstream): ce type de paramètre est vérifié lors de la transmission de message PARTIAL ACCEPTANCE. les nœuds vérifient les paramètres d'un nœud à autre dans un sens inverse de chemin c'est à dire du nœud destinataire au nœud initiateur (source).

3.4 Conclusion :

Le protocole de communication proposé pour la négociation des services Cloud avec garantie de niveau de service définie des structures et des règles simples. L'implémentation du protocole est simple. Le protocole est capable d'inclure tous types de services avec leurs propres paramètres. Le protocole s'exécute en parallèle avec le protocole BGP, ceci permet au fournisseur Cloud de garder leurs configurations tout en incluant le nouveau protocole. Le mécanisme est simple, à la réception ; le nœud vérifie si le message est un message BGP ou un message qui appartient au protocole. Ceci est fait grâce à la structure commune de l'entête des messages.

Le protocole bénéfice de la fiabilité de protocole TCP. Il renforce lui aussi sa fiabilité avec la robuste confirmation des requêtes et des réservations dans une session de demande de service. La négociation de service passe par plusieurs étapes et utilise multiple nouveaux types de message (**Figure 44**). Les points forts du protocole du point de vue conceptuelle ont été décrit et reste à implémenter et estimer ses caractéristiques techniques par rapport à certains paramètres de communication. Nous verrons l'implémentation et l'exploitation du protocole et les résultats tirés depuis les expériences.

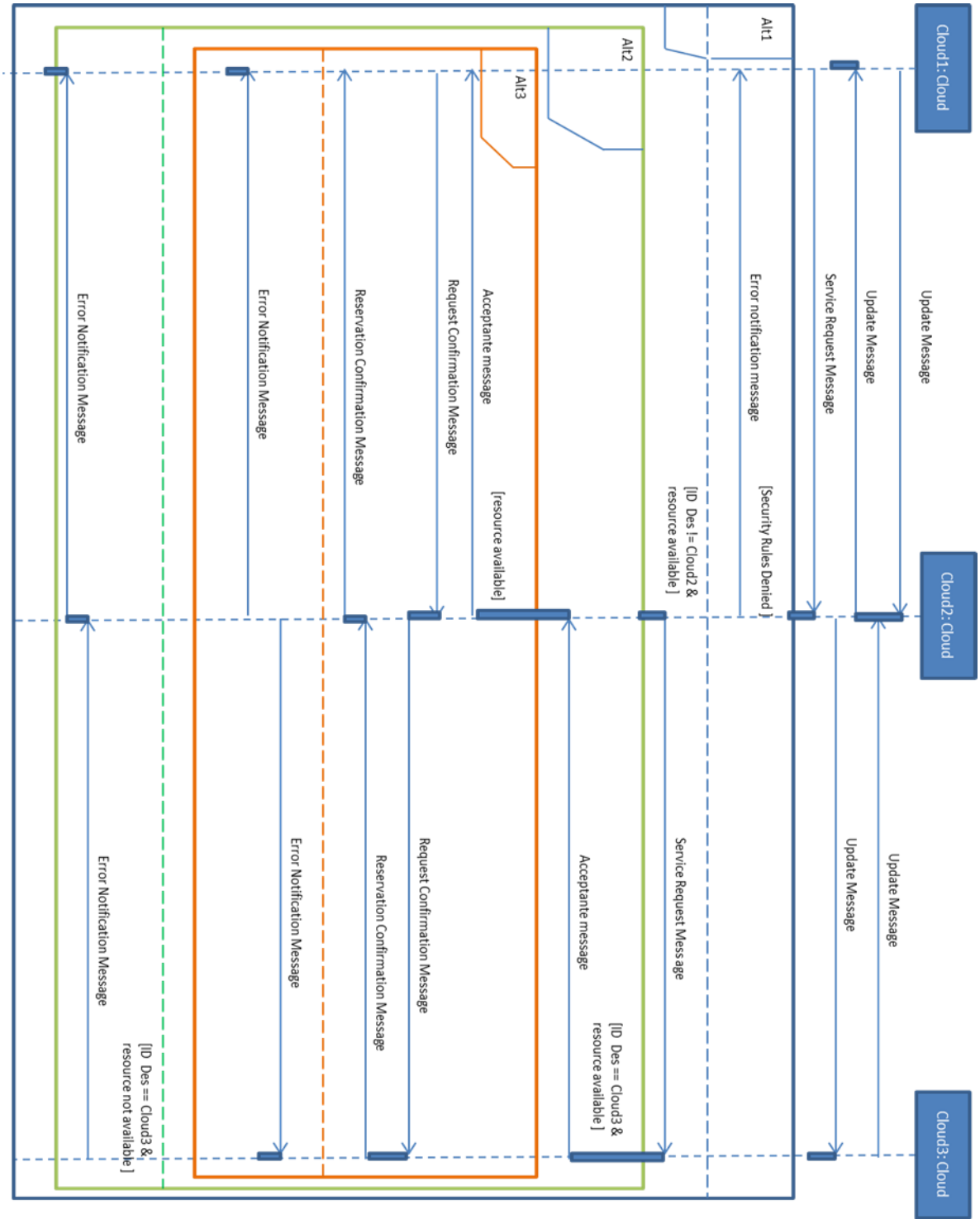

Figure 44 : Diagramme de séquence de protocole proposé

Chapitre 4

Implémentation et résultats

4.1 Introduction

Avec la conception de protocole, nous avons réalisé le protocole dans un environnement très proche de la réalité, c'est-à-dire que le protocole développé est capable d'être exécuté dans un nœud réel (Routeur, Ordinateur ou autre) sans subir des modifications et même si des modifications sont nécessaires, ils ne seront pas majeurs. Nous avons développé le protocole avec le langage bas niveau C. le protocole suit une architecture procédurale. L'environnement d'exécution est l'émulateur NETKIT [86]. L'émulateur NETKIT permet de créer des machines virtuelles contenants seulement le noyau du système d'exploitation Linux et sans interface graphique pour permettre une génération d'un grand nombre possible de machines virtuelles. Nous avons utilisé l'environnement de développement open source Qt [87]. L'ensemble des structures qui définissent les messages et l'enregistrement qui permettent la comparaison et la vérification des identifiants et les requêtes ainsi que les messages. Le protocole nécessite une connaissance de la programmation système de noyau Linux. Nous sommes revenu sur plusieurs techniques de programmation comme l'utilisation des Threads POSIX, Exclusion mutuelle, Sockets, Timer …etc. L'implémentation réelle de protocole nous a permis d'estimer des paramètres réseaux qui peuvent donner une vue sur la performance et le comportement de protocole. Nous essayerons dans ce chapitre de réaliser l'approche d'implémentation et le flux d'exécution du programme dans chaque nœud. Nous discuterons aussi les résultats obtenus. Ensuite, nous discutons les travaux possibles qui peuvent mener à l'amélioration de protocole et les axes que le protocole ouvre.

4.2 Implémentation du protocole :

Le nouveau protocole proposé est développé en langage C dans l'environnement de développement Open Source dans une machine duel Core 2.6 GHz et une taille de mémoire égale à 2 Go. Le système d'exploitation est Ubuntu 14.6 LT. Le même programme s'exécute dans tous les nœuds (machine virtuelle) générées par l'émulateur NETKI.

Principalement, notre programme suit une logique de tout autre protocole réel. Au début de l'exécution de protocole après avoir passé l'identifiant du Cloud et la liste des identifiants des services offerts par le protocole, le protocole commence á réserver les ressources qu'il nécessite pour son fonctionnement. Dans une deuxième étape, le programme lance une suite de threads, chaque type de thread est associé á une interface. La première phase qui correspond á la réservation et initialisation des ressources nécessaires pour le fonctionnement du protocole. L'ensemble des initialisations sont illustrées dans la **Figure 46 :**

- ***Initialisation de la table de routage*** : lors du démarrage du programme de protocole, l'identifiant de Cloud et les services qu'il offre sont entrés comme paramètre de programme. La première fonction convertit ces entrées et crée la table de routage initiale qui va contenir simplement les entrés ID Cloud, ID service et qui seront les premiers à être partagés avec les nœuds voisins.

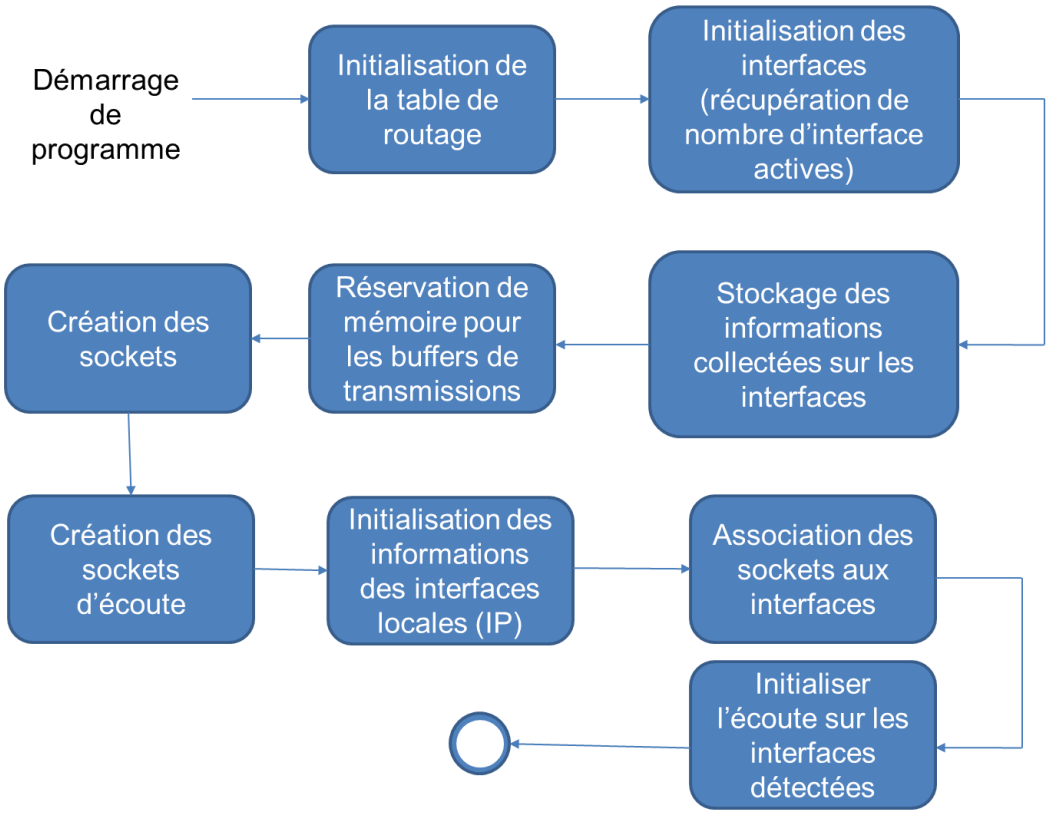

Figure 46 : séquence de réservation et initialisation des ressources

- ***Initialisation des interfaces*** : dans cette partie, on détermine le nombre d'interfaces actives et connectées. En plus de la détermination du nombre d'interfaces, nous gardant cette information dans des tableaux qui nous permettent de les retrouver si nécessaire sans refaire la même procédure.
- ***Réservation des mémoires des buffers*** : dans cette étape, nous réservons la mémoire à tous les buffers nécessaires pour la transmission des données. Chaque buffer est associé á une interface de sortie, lors de la sélection de l'interface de sortie, un thread va mettre les données à transmettre dans le buffer de l'interface de sortie.
- ***Création des sockets*** : après avoir récupérer les informations sur les interfaces, en initialise un nombre de sockets égales au nombre d'interfaces. Deux types de sockets sont initialisés. Les sockets d'écoute qui seront utilisées pour attendre tous les connexions TCP de nœud voisin, le deuxième type sont les sockets de connexion c'est à dire des sockets utilisées pour se connecter aux nœuds distants.
- ***Initialisation des informations des interfaces locales*** : pour pouvoir ecouter sur chaque interface nous devons connaitre l'adresse IP de cette interface. Pour cela nous inserons ces informations dans des structures standards de communication des sockets en langage C.
- ***Association des interfaces aux informations*** : Après l'initialisation des sockets et les informations de chaque interface, nous associons dans cette étape les sockets aux interfaces.

- ***Démarrage d'écoute pour des connexions TCP entrantes*** : la dernière étape de la première partie d'exécution de protocole consiste à démarrer l'écoute sur les interfaces dans le cas où un nœud distant essayera de connecter.

Dans une deuxième étape, nous lançons les threads de traitement. Chaque type de thread est associé à une interface. Il y a cinq types de thread, le premier thread consiste à initialiser une connexion TCP au nœud voisin. Le deuxième thread attend une connexion TCP entrante. Le troisième thread commence à attendre que des messages de négociation de service soient reçus. Un quatrième thread pour l'émission de premier message update. Le quatrième thread envoie en boucle des messages de contrôle (acceptation, requête …etc). Chaque type de thread est dupliqué plusieurs fois. Au début du programme, un nombre X de thread est initialisé. Après, un nombre N de thread vont être exécuté et les autres seront arrêter (X>=N). La **Figure 47** représente un schéma de haut niveau de la deuxième partie de l'exécution de protocole. Nous viendrons avec plus de détails sur ces threads.

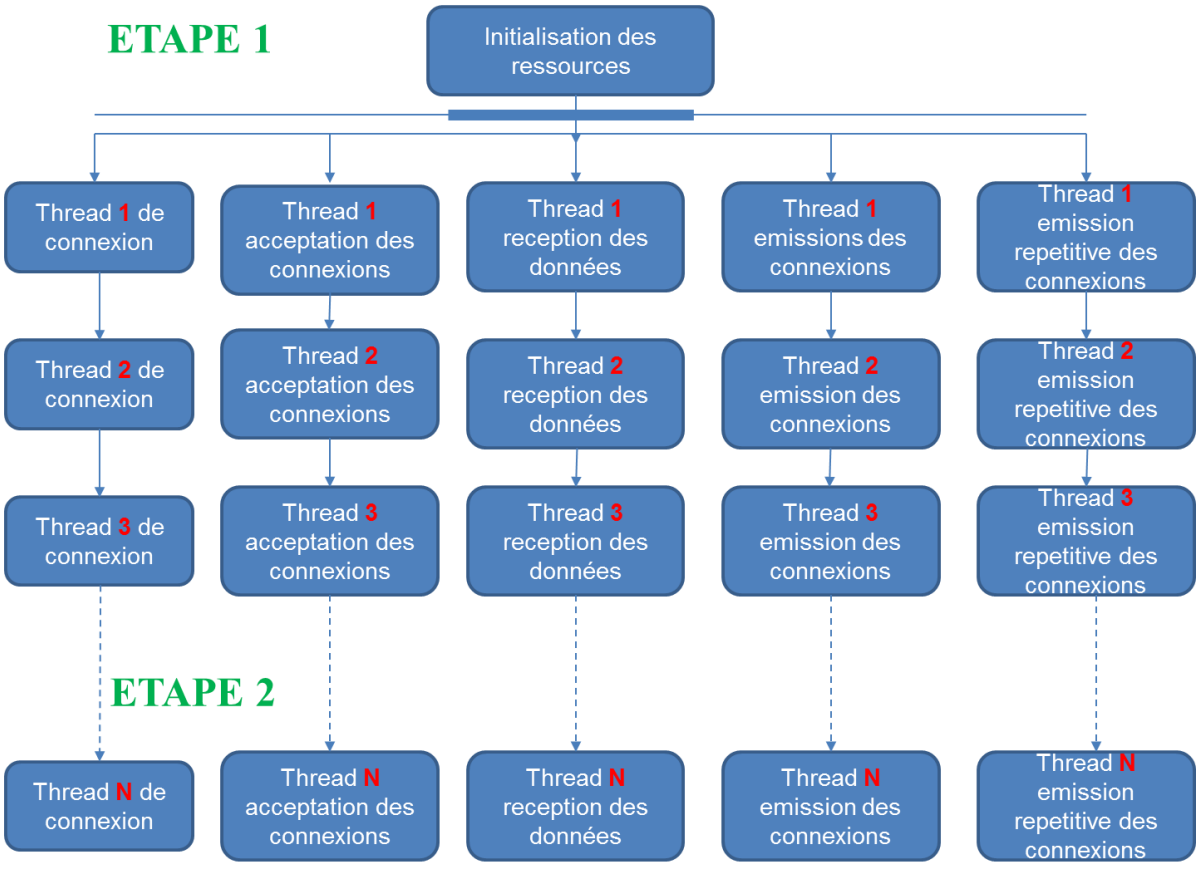

Figure 47 : Lancement des Threads de traitement de protocole

4.2.1 Thread de Connexion :

Le thread de connexion vérifie comme dans tous les autres threads si le nombre d'interfaces est supérieur au numéro de l'interface. Si le nombre de l'interface est supérieur au numéro d'interface calculé au début de la phase d'initialisation des ressources, le thread va quitter totalement l'exécution. Dans le cas où la condition est juste, il essaye de se connecter au

nœud distant en utilisant les informations stockées dans la phase précédente. Le nœud exécute en boucle la fonction de connexion tant qu'il marque un échec dans un temps bien déterminé. L'exécution en boucle considère que le nœud distant peut ne pas avoir lancé l'exécution du programme de protocole en question. Après une connexion avec succès, le thread démarre un trigger au thread responsable de transmission (sans boucle). Ce thread sera fermé après avoir lancé le trigger de thread de transmission.

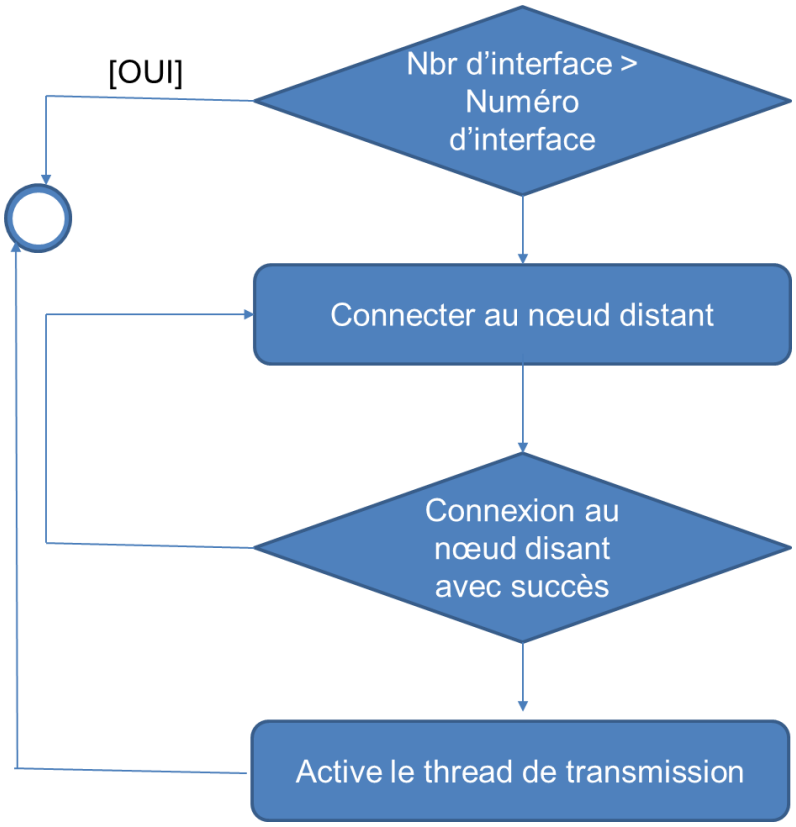

Figure 48 : Diagramme d'exécution de thread de connexion

4.2.2 Thread d'acceptation des connexions :

Le thread d'acceptation des connexions attend qu'une connexion TCP soit demandée par le nœud voisin. Mais avant, il doit vérifier la condition de nombre d'interface contre le numéro d'interface. Si le nombre d'interface est supérieur au numéro d'interface alors le nœud va attendre qu'une connexion TCP soit demandée, il accepte cette connexion et active le thread de réception des données.

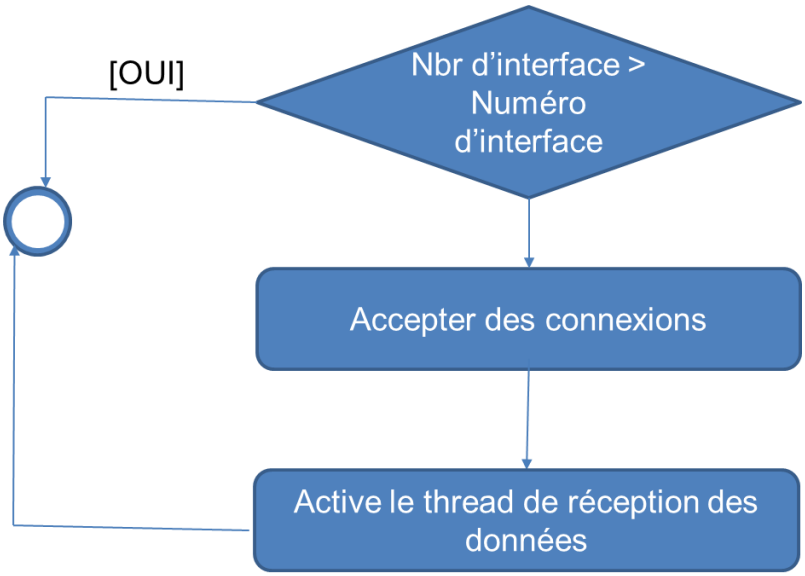

Figure 49 : Diagramme d'exécution de thread d'acceptation

4.2.3 Thread de réception des données (messages de contrôle) :

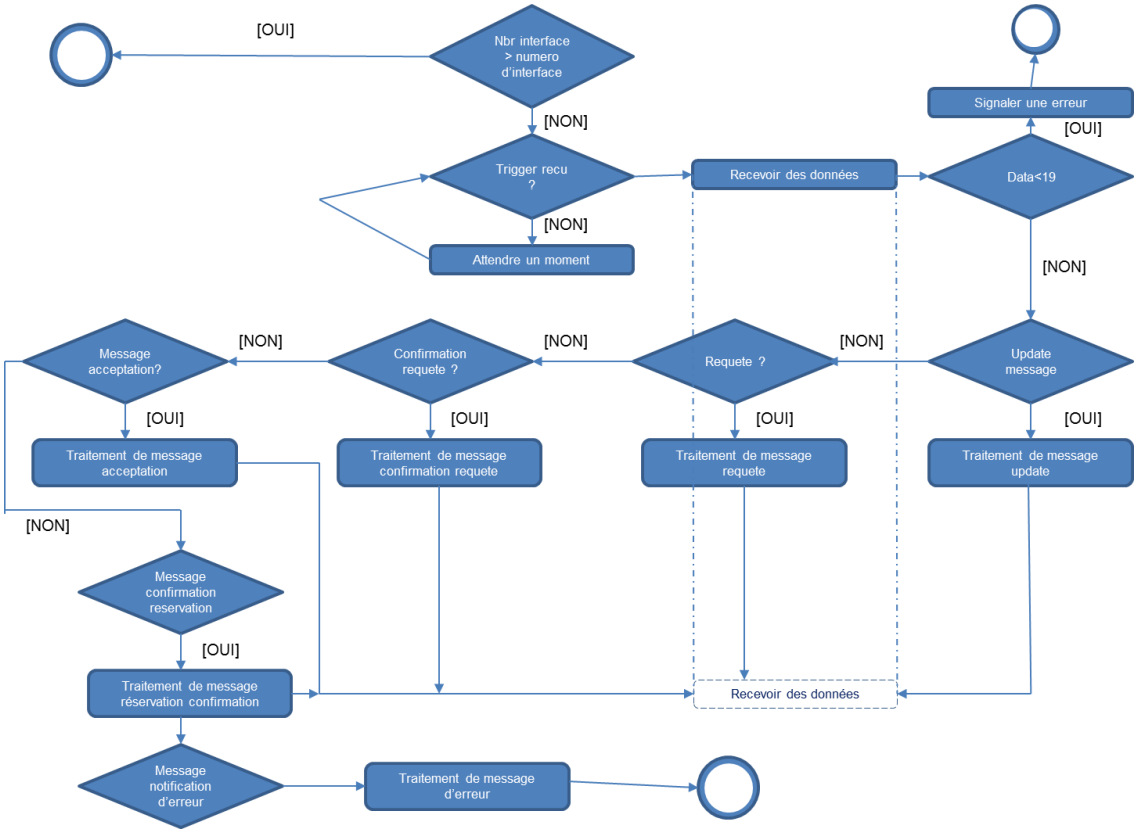

Figure 50 : Diagramme d'exécution de thread de réception des données

Le thread de réception des messages de contrôle vérifie au début si le nœud doit être associé à une interface ou pas en vérifiant le numéro de l'interface contre le nombre d'interface détecté au début de programme. Après cette vérification, on attend que l'évènement d'activation de thread apparaisse. Après la réception de signal de début d'exécution de thread, on reçoit la

première série de données venant du nœud voisin. La première vérification se fait sur la longueur de message. On tire le header et on le décortique pour connaitre la longueur de message et le type. Si le message reçu est inférieur au minimum d'un message de contrôle (entête 19 Bytes) on arrête l'exécution avec envoi de message d'erreur (Non mentionné dans le diagramme). Dans le cas où la taille est plus grande que le minimum d'un message de contrôle, on passe au traitement de message. A partir de l'entête reçue, on vérifie si la taille de message de contrôle correspondant à la longueur indiquée dans l'entête. Après cette vérification on différencie le type de message et on exécute le traitement qui correspond à chaque type de message de contrôle. En cas d'erreur on ferme la session. Dans le cas contraire, on finit le traitement et on exécute en boucle infinie pour la réception et le traitement d'un message.

4.2.4 Thread d'émission des données (non répétitive) :

Le thread attend un signal de trigger pour démarrer l'exécution des actions de transmission de message update lors de la première connexion TCP. Apres la réception de trigger, le nœud vérifie si le thread a une interface á laquelle il doit être associé. En cas de non association disponible, le thread arrête l'exécution définitivement. Dans le cas contraire, le nœud va initier le premier message update. Toutes les transmissions et réceptions des données de contrôle suivent une règle commune. Lors de la transmission, on crée un message non contigu en utilisant des structures complexes, on suit en parse le contenu de la structure (objet) dans un espace mémoire contigu et on transmit. Lors de la réception, on reçoit un message contigu et le parser dans un objet/structure non contigu et on exécute le traitement du structure. Après la transmission du premier update, le thread entre en boucle. Dans cette boucle il vérifie si il y a des données à transmettre ou pas (buffer de transmission initialisé au début). Le buffer est écris par plusieurs threads et lit par un seul thread. L'exclusion mutuelle est utilisée et conditionnée par la présence des données et la libération de d'action d'écriture et de transmission des données présentes dans le buffer.

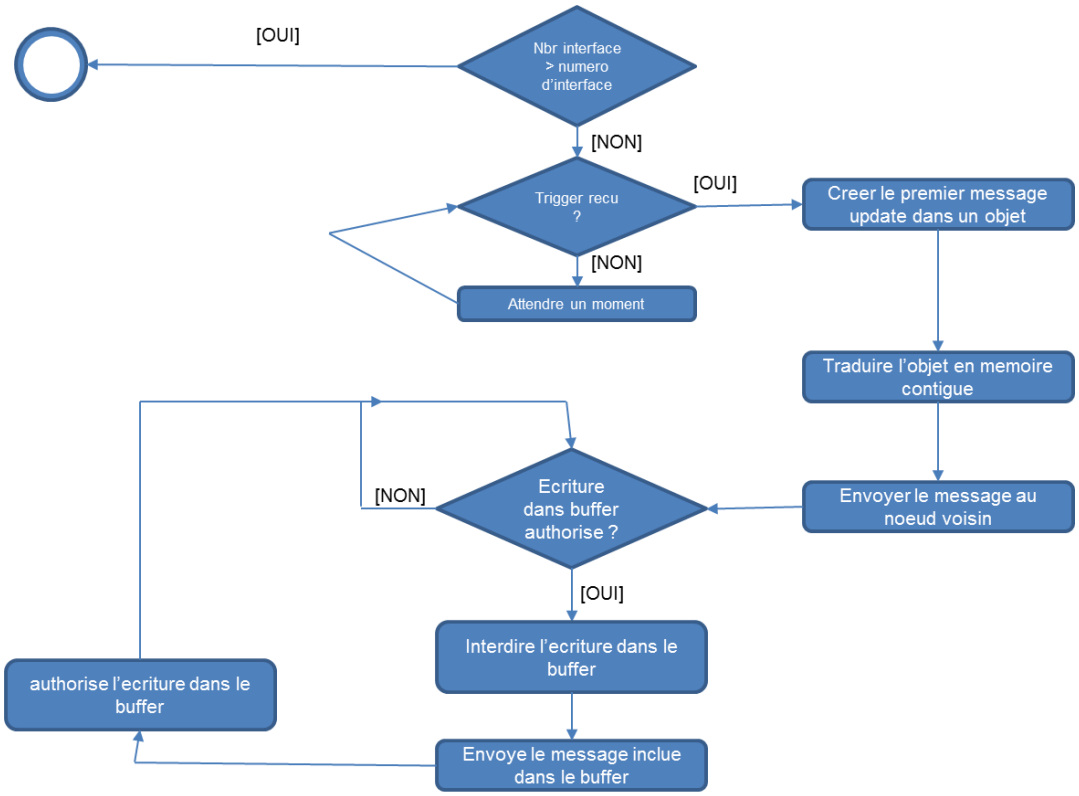

Figure 51 : Diagramme d'exécution de thread transmission (non répétitive)

4.2.5 Thread d'émission des données (Répétitive : message update)

Un dernier type de thread est celui qui essaye d'envoyer répétitivement les messages de mise à jour à chaque fois que de nouvelles routes existent. En respectant le timer de retransmission des messages des paquets. Donc, le nœud attend un signal d'un autre thread précèdent pour démarrer l'exécution. Avant cela, il doit toujours vérifie s'il y a une interface avec laquelle il va s'associer. Si une interface est active, il continue l'exécution sinon il quitte le programme. Dans le cas contraire, il attend l'expiration de timer, si le timer expire alors il vérifie s'il y'a dans la table de routage de nouvelles entrées, il prépare les nouvelles entrées dans un message (Objet) ensuite, il reformate ce message en message contiguë pour qu'il soit prêt à le transmettre. Après la transmission du message, il remet á zéro le timer et attend de nouveau l'expiration, vérification d'existence de nouvelles entrées et ainsi de suite. Dans le cas où le timer expire et qu'aucune entrée n'est ajoutée, le timer sera réinitialisé et toute autre nouvelle entrée qui est enregistrée après ce timer doit attendre de nouveau son expiration. D'une autre façon la décision ne dépend pas de combien de fois le timer a expiré sans envoyer des messages mais juste de l'expiration de timer.

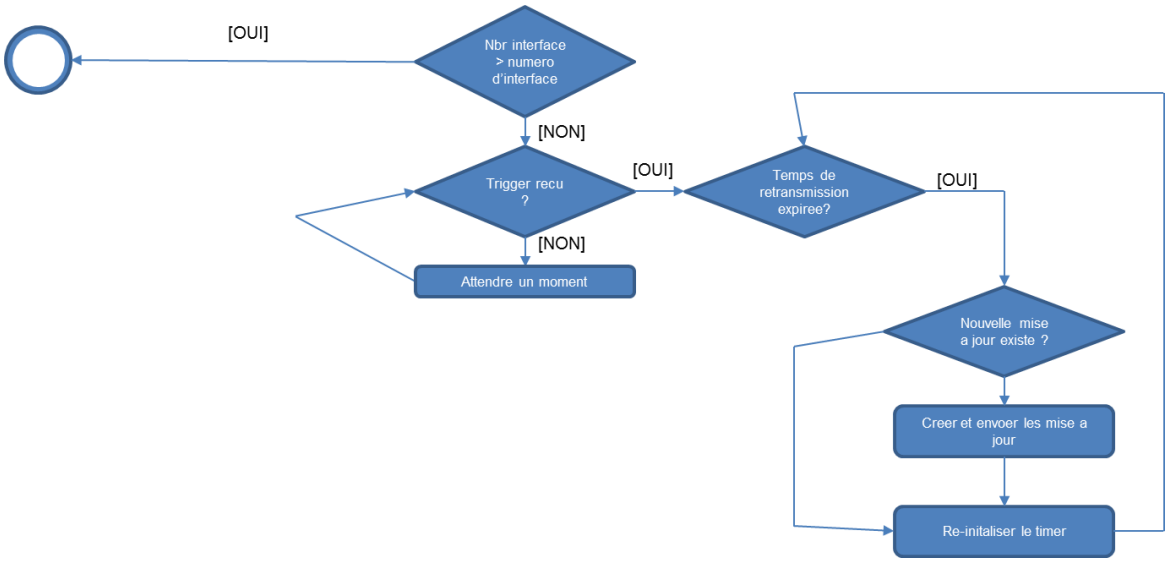

Figure 52 : Diagramme d'exécution de thread de transmission (répétitive updates)

4.3 résultats et discussion :

Dans une première partie, nous avons estimé le temps de convergence de protocole et aussi la quantité de données générées. Le temps de convergence est estimé en changeant le nombre de nœud et la séquence de démarrage des nœuds. Pour le data over Head, nous avons estimé la quantité de donnée en changeant le nombre de services offerts par chaque Cloud. Les deux paramètres réseaux (temps de convergence et quantité de données générées) sont liés au message UPDATE.

4.3.1 Temps de convergence :

Le temps de convergence d'un protocole de routage dans les réseaux informatiques est connu comme un paramètre important dans l'évaluation de la performance d'un protocole de routage. Pour estimer le temps de convergence, nous avons supposé que chaque fournisseur de service Cloud offre un seul service. Nous commençons par deux nœuds seulement dans le réseau, jusqu'aux six nœuds interconnectés et exécutant tous le protocole de routage. De plus, nous avons changé les séquences d'exécution. Un paramètre important qui influe sur le temps de convergence, c'est le choix de la valeur du timer qui autorise un nœud à envoyer des messages de mise à jour UPDATEs. Nous verrons que l'architecture et les séquences d'exécution sont importantes alors nous avons bien mentionné les architectures dans les figures 1, 2, 3, 4 et 5.

Chaque architecture est constituée d'un nombre fixe de nœud (fournisseur de service Cloud). Chaque nœud a un numéro et un nombre d'interface numéroté de 1 à N. les liens entre interface sont respectés dans la configuration des machines virtuelle dans l'émulateur NETKIT. Si on prend l'exemple de nœud 1 et nœud 2 dans la **Figure 53.** Le nœud 1 est connecté au nœud 2 via l'interface 0 de nœud 1 et l'interface 0 de nœud 2.

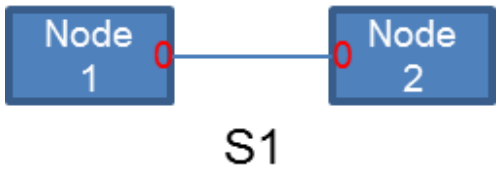

Figure 53 : S1- Architecture Réseau à deux nœuds

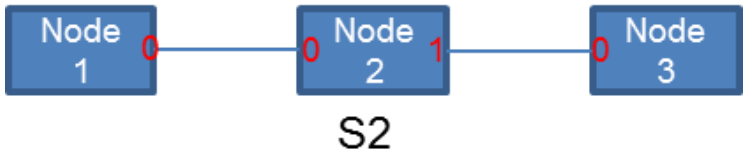

Figure 54 : S2- Architecture Réseau à trois nœuds

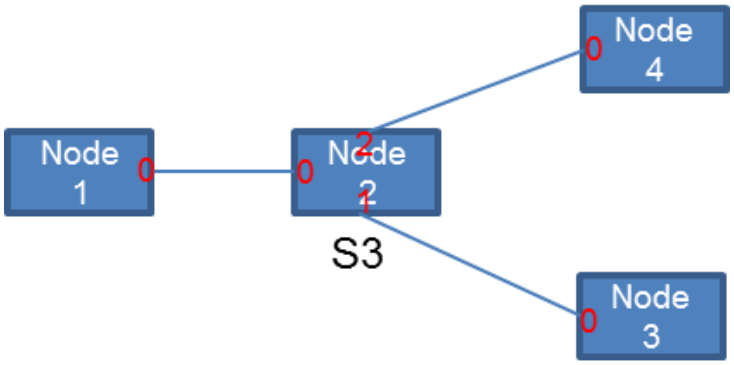

Figure 55 : S3- Architecture Réseau à quatre nœuds

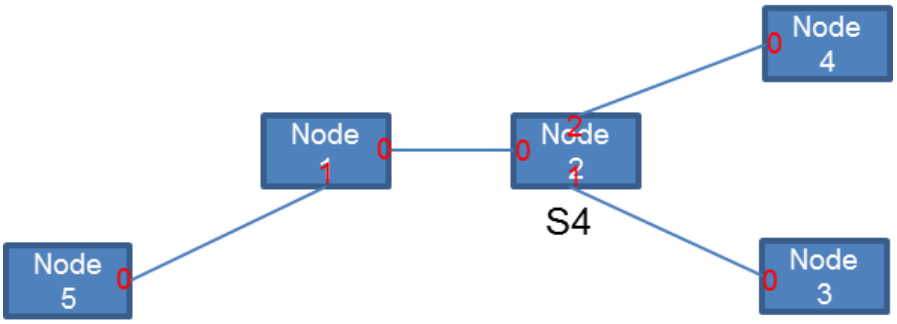

Figure 56 : S4- Architecture Réseau à cinq nœuds

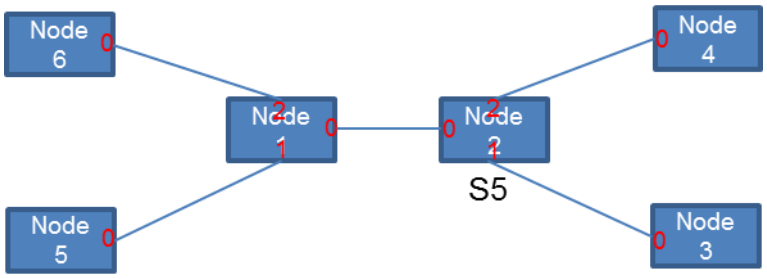

Figure 57 : S6- Architecture Réseau à six nœuds

Les résultats obtenus sont contenus dans le **Tableau 1.** Les résultats de simulation sont pris par moyenne de cinq différentes expériences pour chaque cas. La valeur de chaque cas dans le **Tableau 1**est la moyenne de cinq expériences.

	2 nodes	**3 nodes**	**4 nodes**	**5 nodes**	**6 nodes**
20s (timer)	6.50 s	17.25 s	30.00 s	39.50 s	41.00 s
40s (timer)	5.75 s	23.25 s	33.75 s	66.25 s	86.00 s
60s (timer)	6.25 s	33.00 s	47.75 s	96.75 s	97.50 s
Average	6.17 s	24.50 s	37.17 s	67.50 s	71.83 s

Tableau 1 : Résultats d'expérimentation de temps de convergence de protocole

Pour une meilleur visualisation des résultats nous avons opté a ploter les données sous forme de courbe (**Figure 58**).

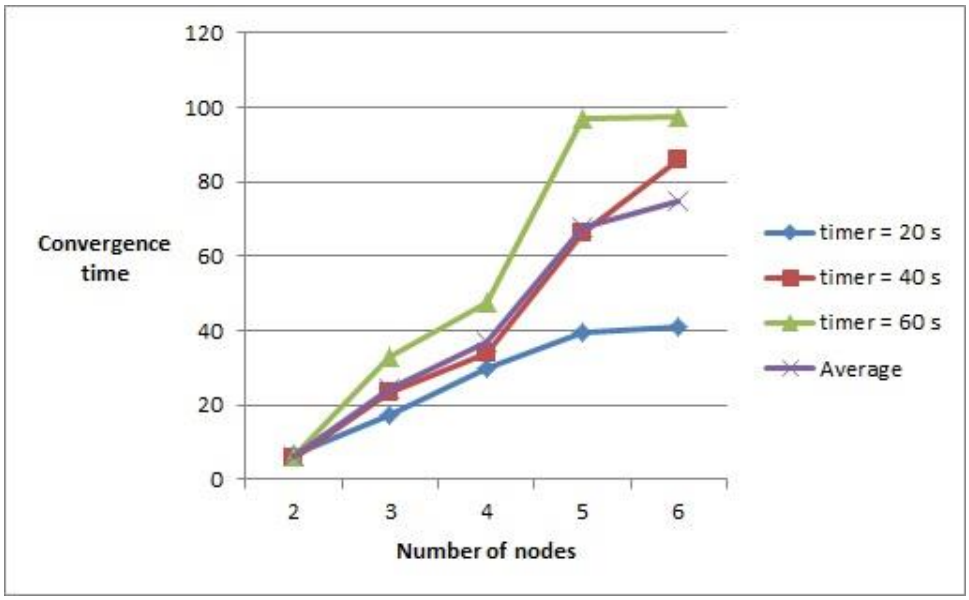

Figure 58 : Courbe des resultats de temps de convergence

Comme résultat concernant le temps de convergence de protocole proposé. **Figure 58** et **Tableau 1** montrent le changement de temps de convergence par rapport au nombre de nœuds présentés dans les figures (s1, s2, s3, s4 et s5). La première information dans les courbes est la relation proportionnelle entre le nombre de nœud et le temps de convergence. Une autre observation c'est que chaque fois que le timer augmente alors le temps de convergence devient plus long. Ces deux relations sont claires et logique. Une troisième

relation qui est liée à la topologie et les nœuds de réseau qui rejoint le backbone des fournisseurs de service Cloud dont leurs séquences est une autre observation à voir.

Concentrant les quatre courbes dans la **Figure 58.** Dans chaque courbe, deux comportements peuvent être distingués. Un petit changement dans le temps de convergence est observé lorsque on passe de trois nœuds à quatre nœuds et de cinq nœuds à six nœuds. Dans le cas contraire, lorsqu'on passe de deux nœuds à trois nœuds et de quatre nœudà cinq nœuds, le temps de convergence montre des valeurs élevées. Ces observations sont claires dans la courbe où le timer est égale á 60 seconds. Le Δt entre le point avec deux nœuds et le point avec trois noeuds est égale à 26.5 seconds et Δt entre le point avec 4 nœuds et le point avec 5 nœuds est égal à 49 seconds. Par contre, Δt entre le point avec trois nœuds et le point avec quatre nœuds est 17.75 seconds et 0.75 seconds entre le point avec 5 nœuds et le point avec 6 noeuds. La raison de ces changements inégaux est l'architecture de réseau des nœuds. D'une autre façon, lorsque la profondeur (le lien le plus long dans l'architecture réseau) du réseau augmente, le temps de convergence devient plus grand que lorsqu'on garde la même profondeur du réseau. Pour référence, la profondeur du réseau change d'un á deux depuis l'architecture S1 (**Figure 53**) et S2 (**Figure 54**) et de deux á trois depuis S3 (**Figure 55**) á S4 (**Figure 56**) respectivement. De ce fait, le temps de convergence augmente beaucoup plus que lorsqu'on garde la même profondeur. En revanche, la profondeur reste la même entre S2 (**Figure 54**) et S3 (**Figure 55**) et qui est égale à deux et de S4 á S5 qui est égale á trois.

La quatrième relation est liée aux séquences de lancement du protocole des nœuds de valeurs vues dans le **Tableau 1**sont la moyennes de plusieurs expériences. Les résultats montraient que lorsque les nœuds qui existent au centre de réseau sont initialisés en premier (comme nœud 1, 2 dans S5) et allant vers les nœuds extérieurs, le temps de convergence sera plus petit que dans le sens contraire.

4.4 Taux de génération des données :

Le deuxième paramètre qui caractérise la performance de notre protocole est bien le taux de génération de données jusqu'à la convergence de protocole. Notre protocole était étudié sur ce point important. Nous avons utilisé une architecture composée de six nœuds (fournisseurs de service Cloud : **Figure 57**). Nous avons exécuté plusieurs expériences, chaque expérience est répétée plusieurs fois. La quantité des données générée est calculée et présentée sous forme de tableau (Tableaux 2). Dans chaque simulation on augmente le nombre de service que chaque fournisseur offre. Le nombre de service offert par chaque fournisseur varie d'un service à six services. En plus du nombre de service on a différencié deux types de simulations en suivant la séquence 1-2-3-4-5-6 et la séquence 3-4-56-1-2. Pour une meilleur visualisions et interprétation du résultat, la représentation en courbe était choisie.

	1 service	**2 services**	**3 services**	**4 services**	**services**	**6 services**
Seq-1-2-3-4-5-6	532 byte	560 byte	993 byte	1032 byte	1071 byte	1110 byte
Seq-3-4-5-6-1-2	646 byte	676 byte	778 byte	854 byte	864 byte	928 byte
Average	589 byte	618 byte	885.5 byte	943 byte	967.5 byte	1019 byte

Tableau 2 : taux de génération des données (convergence)

La **Figure 59**représente trois courbes différentes. La première est le taux de génération de donnée du protocole où la séquence débute les nœuds suivant la séquence 3-4-5-6-1-2 veut dire que le nœud qui commence l'exécution de protocole est le nœud numéro trois, ensuite le nœud numéro quatre ensuite le nœud numéro cinq et ainsi de suite. Même chose pour la courbe seq-1-2-3-4-5-6. La troisième courbe correspond a la moyenne des deux courbes précédentes.

Les courbes montrent trois parties, la première partie est entre l'offre de zéro service á deux services. La deuxième partie est entre deux services et trois services et la troisième et la dernière partie est entre trois et six services. Le plus petit taux de génération des données par le protocole montrée est au environ de 589 Bytes lorsque chaque fournisseur de service Cloud offre seulement un service.

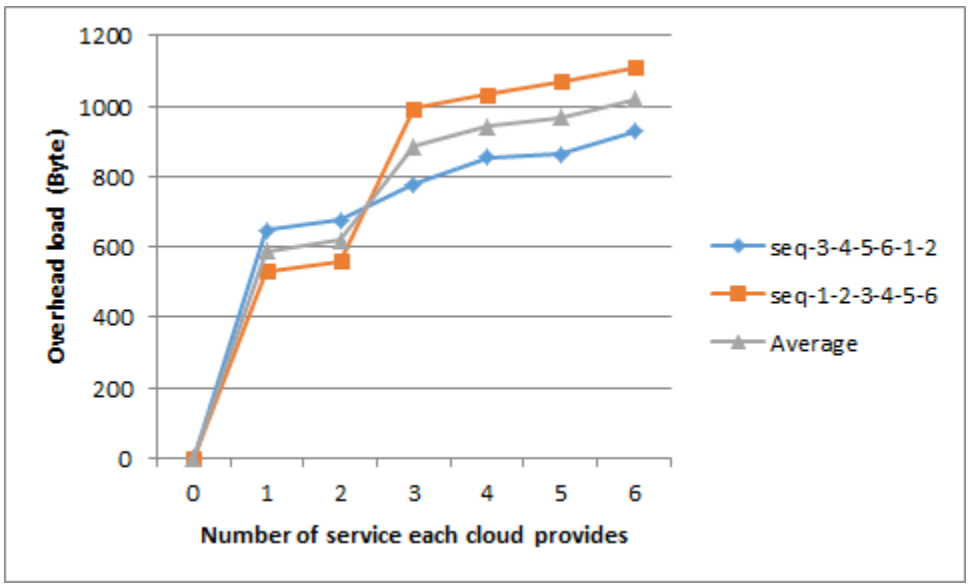

Figure 59 : Taux de génération des données de protocole (Convergence)

Les résultats montrent que lorsqu'on suit la séquence 1-2-3-4-5-6, le protocole génère moins de donnée que lorsqu'on suit la deuxième séquence 3-4-5-6-1-2. Première déduction, commencer l'exécution de protocole depuis les nœuds centraux (nœud 2 et nœud 3) avant les nœuds extérieurs va réduire le taux de génération de données par rapport à la séquence extérieurs vers l'intérieur. Une chose à notifier, c'est qu'en allant d'un seul service á deux services par chaque Cloud, le taux de données générées augmente légèrement de 532 Bytes á 560 Bytes avec une différence de 28 Bytes dans la séquence 1-2-3-4-5-6 et de 646 Bytes á 676 Bytes avec une différence de 30 Bytes. Dans la deuxième partie, un changement claire est observée où une grande augmentation est notifiée dans la séquence 1-2-3-4-5-6 allant de 560 Bytes á 993 Bytes avec une différence de 433 Bytes. Contrairement á la séquence 1-2-3-4-5-6, la séquence 3-4-5-6-1-2 a gardé une augmentation petite dans le taux de génération des données allant de 676 Bytes á 778 Bytes avec une différence de 102 Bytes. Ce deuxième ensemble de résultat est contradictoire avec la première déduction. Mais, ceci a une explication. La raison c'est que la même séquence est gardée mais le temps était changé

lorsque chaque nœud doit commencer. Sinon, dans la deuxième et la troisième partie des courbes, la même séquence était gardée mais un autre nœud commence seulement après la convergence de premier nœud lancé en leurs donnant le temps nécessaire pour converger. Dans cette partie, une autre conclusion est déduite que la séquence de lancements d'exécution de protocole par les nœuds tous seul ne donne pas une information si la charge (taux de donnée générée) sera grand ou petit. La troisième partie montre une bonne tendance de taux de convergence de donnée. La courbe de la séquence 1-2-3-4-5-6 est au-dessus de la courbe de la séquence 3-4-5-6-1-2. Ce résultat donne beaucoup plus d'évidence pour dire que l'exécution de protocole en commençant de l'extérieur de réseau vers l'intérieur de réseau dans un temps suffisant de convergence conduira á une génération de donnée minime. De l'autre côté, une augmentation linéaire apparait quand le nombre de services fournis par chaque nœud augmente. La différence entre chaque étape (augmenter le nombre de service par un) est au environ de 39 Bytes pour la séquence 1-2-3-4-5-6. Dans la séquence 3-4-5-6-1-2, la différence entre chaque étape est 76 Bytes, 10 Bytes et 64 Bytes respectivement.

En général, tant que le nombre de transition entre tous les nœuds est réduit, un paquet UPDATE commun avec un seul entête est créé, de cette façon le taux de génération de données est réduit. Le nombre de transition dépend du nombre des nœuds dans le réseau, le nombre de liens, séquence de lancement et temps de lancement. Il est logique que tant que le nombre de service augmente, le taux de donnée généré augmente mais le protocole montre une augmentation légèrement linéaire par rapport au taux de génération des données. L'expérience montre une scalabilité de protocole qui peut être utilisée pour un réseau de grand rang où le temps de convergence et le taux de génération de données sont acceptables. L'approche est de minimiser les messages update en mettant seulement les champs nécessaires pour qu'elle soit échangée au début de la connexion. N'importe quelle autres informations soient apprise automatiquement par des sessions des requêtes multiples des services. Avec cette stratégie, une remise à jour de table de routage des services est facilement faite dans le cas où un lien tombe en panne ou qu'il devient défaillant et le taux de génération de donnée jusqu'à la convergence sera trop petit.

4.5Conclusion

Toutes les solutions qui se basent sur l'interopérabilité ont besoin d'un protocole de communication qui exprime les opérations et les données et les règles d'un environnement particulier. Notre environnement est de celui de Cloud et pas seulement le Cloud mais une grande sous partie inévitable dans la progression de la technologie Cloud qui est la coopération et l'interopérabilité entre les fournisseurs de services Cloud. Notre travail contribue à un point nécessaire dans ce domaine qui est la standardisation des infrastructures Cloud. Dans presque toutes les recherches rencontrées dans ce domaine, rares qui ont fait une proposition d'un protocole au complet. Notre protocole peut être une bonne initiative pour qu'il soit pris par d'autres chercheurs et le modifier pour répondre à d'autre besoins de système Cloud comme la sécurité. Notre protocole montrait des résultats satisfaisants. Si on voit, il est simple à comprendre et donc à implémenter. Le protocole ne génère pas trop de données de contrôle. Le protocole combine entre des actions réactives et proactives donnant un compromis entre les avantages de chaque approche. Notre expérience dans le

développement des protocoles TCP nous a permis de concevoir un protocole qui répond aux besoins de la communication inter Cloud pour la négociation des services avec exigence de qualité de service. Le temps de convergence lui-même montre un taux faible sachant que le nombre de fournisseurs de Cloud peut augmenter dans les années à venir. Il est important pour nous montrer comment notre protocole peut fonctionner avec n'importe quel type de service qui peut être spécifique à un domaine scientifique particulier. Pour cette raison, nous nous sommes intéressés à un domaine très intéressant pour l'humanité et qui est devenu un domaine des grands. C'est le domaine spatial et plus en particulier les missions d'observation de la terre. Les grandes agences spatiales ont déjà commencé à utiliser le Cloud dans l'exploitation de l'espace. Pour cette raison nous avons appliqué notre protocole dans un exemple qui donnera un argument puissant sur la capacité de protocole d'être employé par n'importe quel domaine.

Chapitre 5

Application

5.1 Introduction

La résolution de problème de collaboration entre différents fournisseurs de service Cloud avec garantie de niveau de service s'avère très important vue les problèmes qu'il peut les résoudre, des problèmes techniques ou financiers. La qualité de service intéresse tous les infrastructures qui ont besoin de ressource IT, peu importe le domaine de leur travail. Parmi les domaines qui ont bénéficié de Cloud et qui peuvent aussi bénéficier de la solution proposée est le domaine de science spatiale. La littérature montre bien comment le modèle Cloud a était adapter par les grandes entreprises comme NASA et ESA. L'Algérie s'est lancée dans le domaine spatial, en particulier dans le domaine d'observation de la terre. De ce fait, notre choix s'est orienté vers ce domaine pour montrer comment notre protocole peut s'adapter à tous les domaines. Avant de s'y lancer dans les résultats d'utilisation des données par le protocole, il est important de présenter ce domaine et les problèmes qui se posent dedans. Nous allons présenter aussi des travaux et des projets de grand importance que les grandes entreprises et sociétés internationales ont optées à faire fonctionner en utilisant le modèle Cloud.

5.2 Le domaine spatial :

Le terme spatial dans notre travail fait référence à l'espace extérieur de l'atmosphère de la terre. Nous pouvons estimer une altitude d'environ 300 Km et plus (reste une référence proposée par nous seulement). La maitrise de l'espace est une nouvelle conquête de l'être humain et les grandes puissances mondiales. L'Algérie pausa sa part dans l'espace avec d'autres pays comme l'Inde, l'Egypte, et aussi les pays développés comme l'USA et le consortium européen ESA. Les missions spatiales suivent un processus de design particulier. Les missions spatiales coutent trop chère. Le cout est l'un des contraintes qui bloque l'ensemble des pays à construire autant de missions. On remarque la collaboration de plusieurs pays pour la réalisation d'un projet spatiale de grande taille qu'il soit des pays du même continent (l'Europe) ou de collaboration internationale (ISS). En plus du cout il y a des contraintes politiques et de risque. Les missions spatiales sont souvent des satellites, Robot ou bien un rover qui sont lancés pour une mission bien déterminée comme l'estimation du champ magnétique de la terre, le taux de pollution de la terre, prise d'image de la terre ou d'une autre planète, pour la communication, l'exploitation des nouvelles objets …etc. l'espace est un domaine très important, pour montrer cette importance, il suffit de poser la question « Que sera notre vie sans la technologie spatiale ? ». On répondant à cette question, nous allons constater qu'il y aura plus de télévision avec de multiple chaines, plus de localisation GPS et alors les navigations maritimes et aéronautique, plus de services téléphonique sans fils dans les endroits isolés …etc. On peut alors imaginer l'importance de la technologie spatiale. Il existe plusieurs catégories de mission. Nous essayons de parler sur deux types de mission plus particulière. Les satellites géostationnaires et les satellites polaires (observation de la terre).

5.2.1 Satellites gestionnaires

Les satellites géostationnaires, sont des satellites qui sont positionnés à une distance de 36000Km loin de la surface de la terre avec un angle proche de 0 degré. Les satellites stationnaires sont souvent utilisés dans des missions de communication. Les satellites

stationnaires pointent toujours sur une région fixe de la terre. Pour garder la même région de couverture, les satellites stationnaires tournent à une vitesse angulaire égale á la vitesse de rotation de la terre (Figure 60).

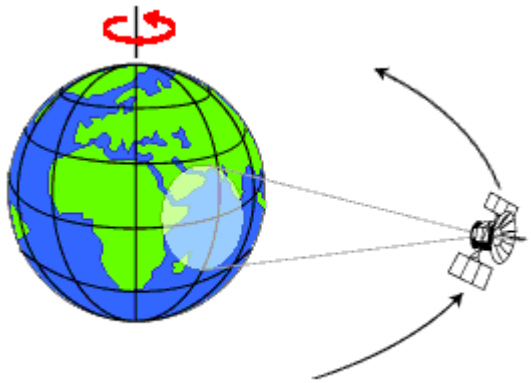

Figure 60: Positionnement d'un satellite géostationnaire par rapport á la terre

La position permanente de satellite géostationnaire donne un avantage de contrôle à partir d'une seule station de sol 24h/24h par contre le satellite pourra seulement couvrir une région et pas la terre entière.

5.2.2 Satellites polaire :

Les satellites qui suivent le chemin d'une orbite polaire sont utilisés souvent dans les études qui concernent la terre. Ces missions peuvent être utilisées pour la collecte des données de pollutions, d'environnement, données scientifiques et plus en particulier dans les prises d'image de la terre. Comme la terre tourne et que l'orbite est fixe par rapport à la référence, le satellite peut couvrir toute la terre pour prendre des images de toutes les régions. La prise d'image sur une région dépendra de temps de passage sur cette région. Si on considère la prise d'image sur une région qu'elle dépend de temps, nous ajouterons aussi que la possibilité de transférer les images vers une station de sol dépend aussi du temps. Ce caractéristique des satellites d'observation de la terre créent un problème de visibilité de satellite par rapport à la station de sol. Les satellites d'observation de la terre peuvent être visibles seulement 30 minutes par jour par rapport à une seule station de contrôle située dans une région proche de la ligne d'équateur. Cette période est l'équivalent de 2.08% de chaque jour (24h) contrairement au satellite géostationnaires qui font une visibilité de 100%. Il est à connaitre que le prix d'une station de sol par rapport à une mission spatial de type observation de la terre peut atteindre la moitié du budget total de la mission. En remarque une défaillance sur le point d'optimisation. Des grandes sommes sont investies et que le matériel n'est utilisé que 2% de la durée de vie de satellite. Ceci est le premier problème qui se pose. Une deuxième chose est celle de traitement des données. Il est connu que la taille des images satellitaire est de taille très grande qui peut dépasser les 2 Go. Le nombre d'image que le satellite peut fournir durant la vie de la mission ne peut être déterminé précisément. Le design des capacités de stockage et de traitement dépend de ce nombre. Comme ce nombre est inconnu, le dimensionnement des capacités de stockage et de traitement des produits image se fait par rapport au pic. Durant la vie de la mission ce pic peut être jamais atteint comme il peut être quasiment dépassé. Ceci est le deuxième problème qu'on le signal. La raison pour laquelle la valeur de pic n'est pas valide car la durée de mission n'est pas prédictible á 100%. Si la durée

de vie augmente, le taux des images peut augmenter. Dans le cas contraire, si la mission aura une durée de vie plus courte que la durée considérée le nombre d'image ne va pas atteindre la valeur maximum définie. Dans le premier cas, il n'y aura pas de place pour le stockage sinon, dans le deuxième cas, les ressources ne seront pas utilisées pour la mission et donc dans les deux cas une erreur de design qui causera des perturbations dans l'investissement et l'optimisation.

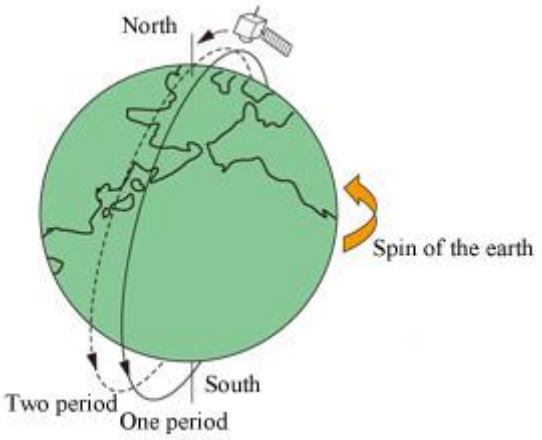

Figure 61 : Positionnement d'un satellite polaire par rapport á la terre ; http://www.eorc.jaxa.jp/en/hatoyama/experience/rm_kiso/satellit_type_orbit_e.html

Comme les satellites des missions d'observation de la terre ont une visibilité très courte, ils sont conçus pour être beaucoup plus autonomes. Les satellites sont donc programmés á effectuer plusieurs actions pour une longue durée. Durant cette période, un évènement non envisager par les actions programmées préalablement peut subvenir. L'évènement peut être de grand risque sur le satellite. Par exemple, il peut y avoir une alerte de débris qui percutera le satellite et faire fin á sa vie en orbite. Dans tel cas, les propriétaires de la mission doivent intervenir plus vite avant que l'acte se produit. Avec une seule station de contrôle disponible, nous devons attendre jusqu'au prochain passage de satellite au-dessus de la station de contrôle. Le temps d'attente peut être non tolérable. Dans ce cas, il faut faire référence à une autre station de contrôle manuellement. Il devient important d'en avoir plusieurs stations de contrôle distribuées dans des lieux distingues de la planète. La solution d'avoir plusieurs stations de sol est contrait par des contraintes politiques et de prix. L'installation d'une station de sol dans un autre pays nécessite des relations politiques. En plus, si le prix d'une station de sol peut couter la moitié de budget mission, on peut imaginer le cout élever de penser à installer plusieurs station de sol. En rapport avec notre contribution, l'application de notre protocole dans ce problème peut servir à montrer la possibilité d'adapter le protocole sur différent domaine et en particulier proposer une solution de collaboration entre agences spatiales pour minimiser les couts et les risques.

5.3 Le spatial et le Cloud

Notre intention d'utiliser le Cloud dans les infrastructures de traitement des données et de communication n'était pas une première sachant que de multiples organisations qui exercent dans différents domaines scientifiques ont adapté le Cloud pour leurs besoins. Les organisations spéciales sous la coupe de l'agence américaine NASA et l'agence européenne

ESA. Penser a utilisé le Cloud a fait aussi l'idée de petite agence spatiale comme celle de l'agence spatiale égyptienne. Les travaux de recherche se lancent pour étudier cette possibilité et de penser à une adaptation sur des aspects différents.

5.3.1 La mission MARS Rovers (NASA) :

NASA avec détermination a décidé d'opérer sa mission Rovers avec une infrastructure 100% Cloud. Elle a conçu des logiciels et des données basés sur le Cloud pour planifier jour en jour les opérations de la mission Rovers.

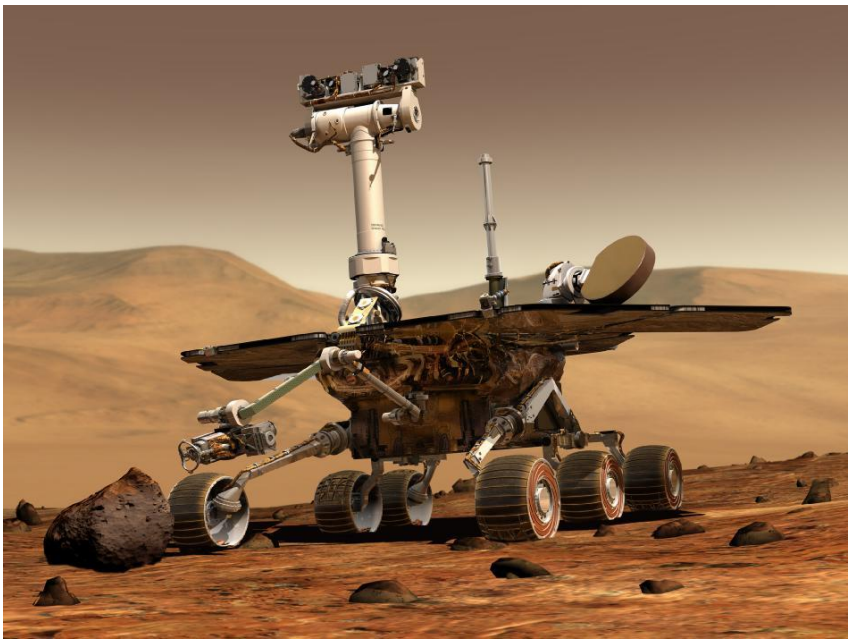

Figure 62 : Le robot Rover ; https://en.wikipedia.org/wiki/Mars_Exploration_Rover

JPL (Jet Propulsion Lab) une partie des départements de NASA a opté pour une solution Cloud pour la mission Rovers. JPL disait avoir gagné de la confiance dans le modèle Cloud Computing. Lorsque la mission a besoin de ressource, elle n'a pas besoin d'installer des nouveaux serveurs. Il suffit de louer depuis le Cloud juste pour le temps nécessaire. Cette solution va permettre au JPL de ne pas trop mêler de consommation électriques des serveurs, la maintenance et le refroidissement pour des serveurs en état idle et le plus souvent, ils ne sont pas utilisés. Grace au Cloud, il devient possible de stocker des données quelque part et récupérer ces données n'importe où et quand c'est voulu juste avec une connexion internet. JPL utilise la nouvelle infrastructure pour exploiter deux missions. Le Cloud leurs a permis d'étendre les capacités pour le contrôle des deux robots [86]. Les opérations des robots nécessitent des plans quotidiennement avec la collaboration des opérateurs et des scientifiques localisés dans des lieux différents de L' USA et l'Europe.

Khwadjashams un ingénieur logiciel dans NASA et auteur de plusieurs articles sur ce sujet disait que le projet Rovers est bien adapté au Cloud Computing. Il ajoute ; le Cloud a permis de délivrer des données à chaque utilisateur inclue dans le projet pour avoir une réaction rapide. La durée de vie de la mission génère une grande quantité de donnée qui était bien servi par la solution Cloud pour le traitement et le stockage des données. Cette capacité infinie de ressource a rendu le Cloud Computing attractif.

Pour la mise en œuvre de l'infrastructure, NASA a collaboré avec plusieurs leaders de domaine Cloud comme Google, Amazon et MS Windows Azure.

NASA dépense 1.5 billions de dollars chaque année sur les systèmes d'informations. NASA a 550 systèmes d'information utilisés par des centaines de milliers de personnes quotidiennement. NASA trouve que le Cloud Computing est plus convenable pour gérer ses infrastructures ITs vue qu'il réduit le cout á travers sa rapidité de déploiement des ressources et améliore la compatibilité pour la coopération. NASA conseil ses agences de passer au modèle Cloud à partir de 2011. La fin 2014, les agences seront seulement capable d'utiliser une plateforme Cloud. En 2009, NASA a réalisé sa première datacenter basé sur une solution Cloud nommée Nebula [87]. Nebula était arrêtée après avoir était comparée avec les capacités de Amazon et Microsoft. Les infrastructures Cloud publics étaient moins chères et fiables. Le grand problème qui faisait fasse et celui de la sécurité. Les données sont considérées de haute sensibilité. En dépit de ce problème, NASA a toujours voulue continuer á progresser dans l'idée de passer vers le Cloud Computing.

Les images venant des missions Mars sont de taille énorme. Les produits d'images sont reproduits depuis les images brutes. La taille des images produite se multiplie et le traitement de ces images nécessite des ressources de calcul énorme. L'un des services des fournisseurs Cloud sont le stockage et le traitement. Ces services peuvent être acquis depuis des fournisseurs Clouds publics. De cette façon, l'accessibilité sera depuis n'importe où en connectant via internet. En plus, la collaboration entre les agences spatiales internationales et que chaque agence peut payer le service qu'elle a utilisée.

5.3.2 Cloud Computing chez ESA

ESA essaie de suivre son homologue NASA dans l'utilisation de la technologie Cloud dans les opérations des missions spatiales. De même, si la NASA s'est orientée vers les fournisseurs publics comme Amazon et Google, ESA a fait appel à la société Orange pour la gestion de Cloud privé d'ESA. Pour répondre à ces besoins, NASA doit avoir de grand nombre de ressources.

ESA conduisait une recherche pour analyser comment le Cloud peut être utilisée pour adresser quelques problèmes des systèmes sol. L'objectif d'étude était sur l'identification des systèmes de données de station de sol pour chaque modèle de provisionnement de Cloud Computing.

Le travail consiste á étudier la façon dont l'infrastructure IT s'étend sur le Cloud. L'étude des solutions de provisionnement open source et commerciaux pour établir un Cloud privé. Un dessin de plan de passage vers le Cloud était évalué sur la capacité de faire évoluer le système de virtualisation classique existant vers une infrastructure virtuelle mais qui se base sur le modèle Cloud Computing. Le travail présenté par l'auteur (Dr. Mehran Sarkarati1 et al, 2012) parle d'objectif de savoir comment les systèmes sol bénéficient de Cloud Computing dans les trois couches (Infrastructure, Plateforme et Software) [88]. Concernant le point d'identification des composants candidats du système sol qui peuvent être adapté au Cloud, l'étude aboutissait à identifier deux composants de la station de sol d'ESA ; Celui de simulateur et le système d'opération d'une mission spatiale dont le nom n'était pas révélé.

Concernant le deuxième objectif de travail de recherche, ESA a pu identifier les composants et les modèles de provisionnement qui les adapte. Nous trouvons le simulateur qui était trouvé adaptable pour la couche plateforme as a service.

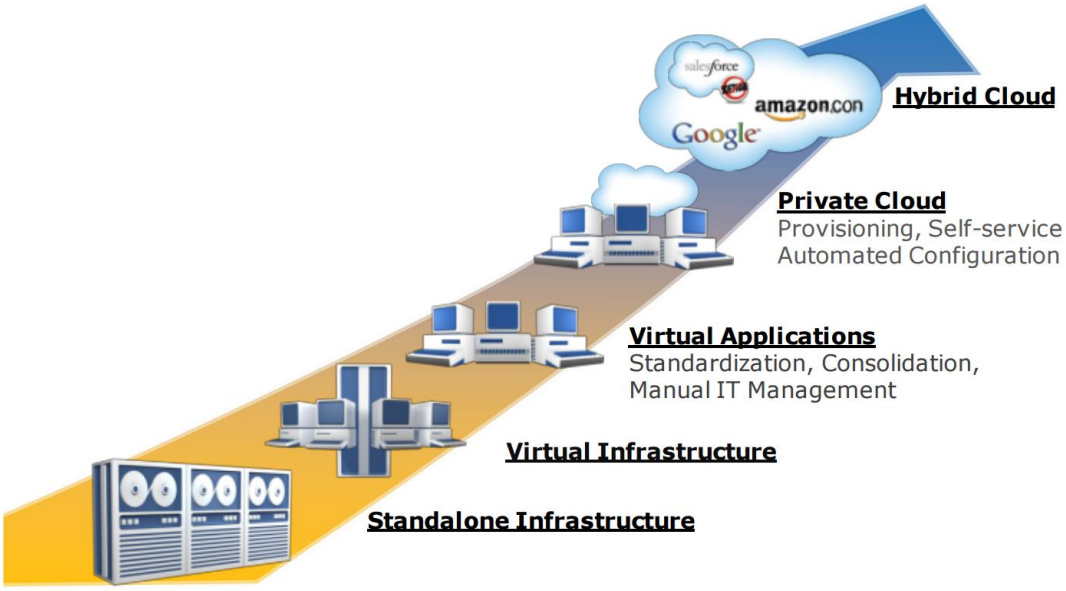

Figure 63 : Plan d'évolution de l'infrastructure du système de sol d'ESA

Les composants du SCOS2000 (Satellite Contrôle and Operations System 2000) comme les paquets, paramètres, structure. Des fichiers ont été identifiés comme adaptable sous forme des softwares dans la couche Software as a Service.

Le service du Climat était identifié comme un PaaS (Portal as a Service). La solution nécessite un passage du manuel á l'automatique. Depuis des années, ESA utilisait la technologie de virtualisation sauf que les requêtes se font manuellement par l'utilisateur et l'administrateur du réseau IT.

Le provisionnement des applications identifiées comme plateforme as a service et Software as a service qui ajoutent autres composants identifiés par rapport au mode de provisionnement de modèle Cloud Computing.

Le plan de passage au modèle Cloud (Figure 63) au sein d'ESA nécessite la détermination des solutions à utiliser pour la création de l'infrastructure Cloud. ESA a évalué suivant 24 critères. Parmi les critères il y avait la possibilité d'intégration dans la structure ESA classique, la sécurité, le support des standard … etc. parmi les solutions open source pour le provisionnement des structures Cloud il y avait OpenStack, OpenNubela, Eucaliptus et VMware vCloud. OpenNebula était l'outil classé en premier dans les solutions open source par contre VMWarevCloud était la solution commerciale qui avait plus d'avantage car l'infrastructure ESA utilisait déjà dans son système de virtualisation des solutions du même vendeur. Le concept de preuve suit les taches suivantes pour montrer la faisabilité avec les outils sélectionnés.

- Provisionnent dynamique des machines virtuelles en ligne de commande.
- De-Provisionnement dynamique des machines virtuelles en ligne de commande.

- Provisionnement dynamique des machines virtuelles avec une interface graphique web
- De- Provisionnement dynamique des machines virtuelles avec une interface graphique web
- Importation d'une nouvelle machine virtuelle en ligne de commande
- Définition d'une pile d'application avec un ensemble de machines virtuelles
- Provisionnement de plusieurs machines virtuelles comme une seule pile d'application
- Une configuration hybride entre un Cloud publique (Amazon) pour le Cloud Computing.

En 201, ESA a eu sa premiere grande infrastructure as a service. De plus, ESA a migré quelque applications vers le Cloud publique. (MehranSarkarati, 2014) présente dans son article scientifique des exemples sur les activités les plus récentes á ESA dans le but de voir comment les couches Plateforme as a service et Software as a Service peuvent aider á résoudre des problèmes et des challenges des logiciels d'opérations des missions spatiales en incorporant un design Cloud Computing dans l'architecture principale[89].

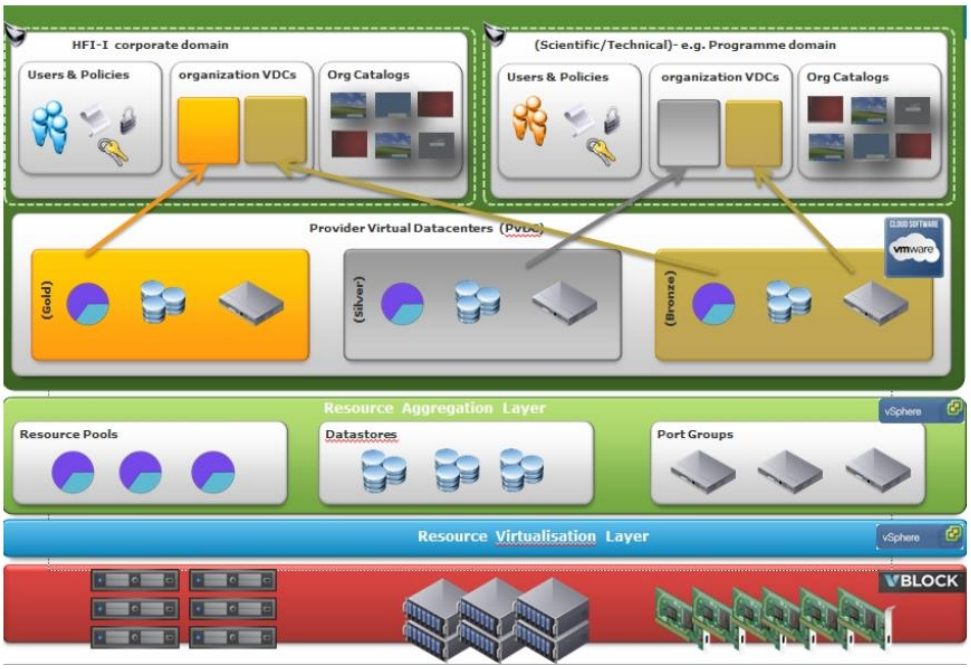

Figure 64: Architecture de solution Cloud pour les missions spatiales

La première initiative d'ESA pour l'utilisation de Cloud vise deux points essentiels :

- Etablissement d'un IaaS privé pour l'offre des ressources virtuelles
- Migration des applications existantes au Cloud privé ou au Cloud public.

Dans les deux cas, le système de provisionnement a une importance pour le déploiement des solutions existantes dans le nouveau modèle (Cloud).

Le Cloud privé (IaaS) est un Cloud interne á ESA sécurisé et multi-tenants qui réserve des ressources de calcul virtuelles (CPU, stockage et réseau) vers un data center virtuel qui les expose aux utilisateurs internes de l'organisation ESA à travers un portail web d'une façon automatisé et établie physiquement dans deux établissement ESRIN et ESOC.

L'infrastructure est basée sur l'outil commercial VBLOCK dans plusieurs racks. La gestion est basée sur l'outil commerciale aussi VMware Entreprise vClouddirector. Alors la virtualisation des ressources est basée sur l'outil VmwareVsphere. vNetwork est un autre outil du même vendeur qui gère les réseaux virtuels. En plus de l'installation et configuration de Cloud privé, ESA a établi un SLA avec un fournisseur de service Cloud public européen.

ESA a fait une analyse sur l'aspect sécurité, où une grande quantité de données sensibles qui s'installent dans le Cloud Computing. Des techniques ont été analysés comme la cryptographie et autres :

- Quatre scenarios sur la sécurité de traitement des données sensibles ont été faits
- Une analyse des risques pour chaque scenario
- Un ensemble d'outils commerciaux étaient identifiés et des métriques sont définies pour l'évaluation des outils.
- Quatre concepts de preuve ont été implémentés pour chaque scenario.

La grande importance que ESA a donné á la technologie Cloud, elle a contribué largement dans les travaux de recherche et des projets Cloud de grande envergure. ESA conduit le projet HelixNubella (un Cloud Scientifique Européen). Le projet inter-partenarial fournira des grandes ressources pour tous les scientifiques et chercheurs européens dans les différents domaines

En parallèle, ESA s'initie dans d'autres travaux Cloud dont :

- GAIA/AGIA : traitement des données scientifiques dans le Cloud publique Amazon
- EOP'S GPOD : grille de calcul pour les applications d'observation de la terre.
- Service de communiquions d'ESA utilisant des solutions des réseaux sociaux.
- Outils de calibration comme web-ex (Cisco - SaaS)

5.3.3 Autre travaux de recherche d'utilisation de Cloud dans le spatial

Autres travaux sont faits en relation avec la science spatiale en utilisant de Cloud Computing. (Steven Johson et al, 2013) présente dans son travail un design et l'implémentation d'une solution basée sur le Cloud Computing pour le projet (SpaceSituationalAwareness) [291]. SSA est un projet à pour objectif de suivre des objets ou des débits dans l'espace et permettre aux opérateurs de savoir s'il est nécessaire de faire des manœuvres ou pas pour éviter une collision. L'auteur essaye de montrer comment le Cloud Computing peut répondre aux challenges du SSA. Il décrit aussi l'architecture proposée.

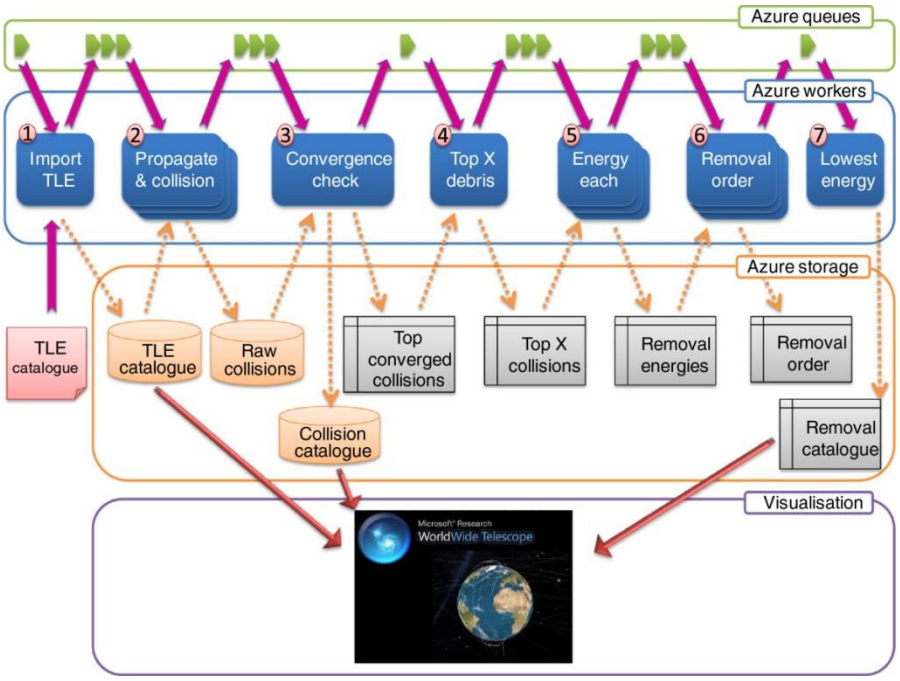

Figure 65 : Architecture modulaire d'une application spatiale

Le projet SSA envisage trois segments principaux :

- La surveillance et le suivit des objets d'humain
- Monitoring de climat et la météo de la terre
- Surveillance et suivit des objets proches de la terre.

L'auteur voit la force d'utilisation de Cloud Computing pour le projet SSA dans quatre points :

- **Diffusion des données** : ESA utilise le catalogue de NASA. Grace au projet SSA, ESA peut offrir le même service. Les capteurs utilisés sont sensibles á des objets de très petite taille. Ceci augmentera le nombre d'objets dans le catalogue. Une collision de deux grands objets peuvent générer plusieurs milliers d'objets de taille plus grande que 10 cm et un nombre qui dépasse les dizaines de milliers de fragments plus grand d'un (1) cm. Vu la nécessité de détection de possibles collisions trop tôt est important, une solution Cloud peut répondre aux besoins de stockage qui offrira une solution excellente pour stocker ce grand nombre d'augmentation de données. L'avantage du stockage dans le Cloud c'est la possibilité de partager ces données entre les partenaires et de coexister des données et faire des calculs dans différent endroits. Les opérateurs peuvent avoir des données depuis le catalogue et sélectionne seulement les objets qui leurs intéressent pour analyser et faire leur calcul de probabilité de collision et puis, chaque partenaire payera pour les ressources utilisée de sa part.
- **Capacité d'expansion** : chaque objet dans le catalogue nécessite un traitement (des analyses d'éventuelle collision : deux á deux). Tant que le catalogue augmente, la demande de puissance de calcul augmente. Le Cloud peut permettre de faire face à la demande brusque des ressources dans un temps réduit. Lorsque le nombre de débris diminue les ressources seront libérées, qui engendre un gain

en termes de coût. la capacité d'extension dans le Cloud offre une possibilité d'augmentation ou la diminution selon la demande pour le projet SSA.

- **Scalabilié** : le catalogue actuel comprend environ 20 000 objets. Avec l'avancement technologique, il devient possible de suivre des objets plus petit que d'un (1) cm. En cas où le Cloud peut s'adapter á ces changements en achetant des ressources supplémentaires. Le SSA peut être scalabale et auto adaptable durant sa période de fonctionnement en utilisant le Cloud Computing.
- **Développement des algorithmes** : les algorithmes complexes des analyses de détection de collision peuvent êtres accélérés en utilisant plus de ressources. á titre d'exemple, il devient possible d'exécuter deux propagateurs en parallèle et comparer les résultats ou les probabilités de collision de chaque propagateur.

Une architecture était présentée (**Figure 65**) qui propose une composition d'un ensemble de modules pour neutraliser et éliminer les débris. L'ensemble des algorithmes et calcul ainsi que le stockage se fait dans un cloud public de Microsoft Azure.

Dans le même contexte des solutions SSA, un autre travail de (Bingwei Liu, 2013) qui évoque l'utilisation de modèle Cloud Computing dans le domaine spatiale et plus en particulier les projets concernés par le SSA [91]. Le travail propose un système d'information fusionnelle pour le SSA basé sur le Cloud Computing. Il examine un prototype qui servira les algorithmes de suivit (tracking). L'auteur discute les bénéfices d'utilisation de Cloud comme alternative aux traitements des données et stockage.

Il y a 1046 satellites qui orbitent autour de la terre (2012). L'auteur revient sur les mêmes points que le projet SSA a besoin dont la grande quantité de stockage, des ordinateurs et calcule puissants et la capacité de traitement et d'accès fiable aux centaines de milliers d'utilisateurs. Dans le futur, les applications spatiales de télédétection peuvent être déployées dans le Cloud.

Plusieurs missions de différents types comme des satellites, des radars et autres capteurs en propriété de différentes agences spatiales qui collectent les données et les envoient vers un Cloud commun. Les utilisateurs (Développeurs, ingénieurs, operateurs...etc.) peuvent ensuite avoir ces données via des APIs. Au-dessus des APIs, il y'a la couche APPS faite pour plusieurs cas comme l'évitement des collisions, détermination d'orbite …etc. L'accès aux services et aux APIs ce fait grâce á un portail web.

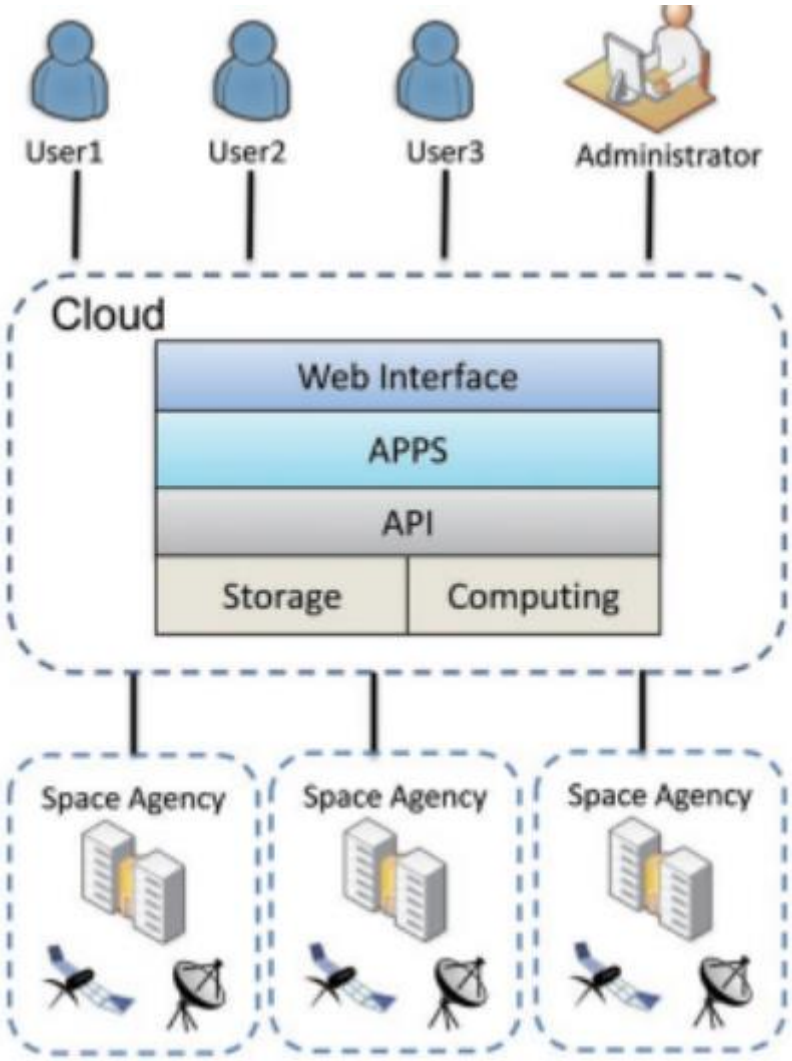

Figure 66 : Scenario typique d'un système SSA basé sur le Cloud Computing

Du coté Cloud, comme il est illustré dans la **Figure 66** qui découpe les ressources en Clusters composés de plusieurs machines virtuelles et physiques. Les clusters permettent d'isoler les taches et les données de chaque machine virtuelle. De plus, les supports de stockage et eux aussi sont isolés et sécurisés avec des technologies comme le service d'intégrité. Les fonctionnalités peuvent être distribuées sur les machines virtuelles. Ceci est une méthode efficace pour implémenter un système complexe. Dans le cas de défaillance d'une machine virtuelle, une autre nouvelle machine virtuelle la remplace. La sécurité revient toujours dans chaque travail qui aborde le modèle Cloud Computing. Trois méthodes sont décrites dont l'isolation des machines virtuelles, la sécurité de stockage et la gestion du réseau.

Du coté spatiale, il y a deux challenges principales. Le premier, est celui du manque des standards, car chaque agence spatiale utilise ses propres protocoles et méthodes. Ceci rend la coordination difficile. Le deuxième point et celui de délai de communication, il est connu que les liens RF n'ont pas le même débit que sur le sol. Ce délai rend la création des applications temps réel est quasiment impossible.

L'auteur a bien mené une expérience de l'architecture proposée avec 16 serveurs dans un data center. Il utilisait la plateforme Xen. Chaque serveur est équipé de deux processeurs Xeon

E5405. Une mémoire de 32Gb et comme capacité de stockage de 3Tb. Un serveur plus puisant était utilisé. Un autre serveur plus puissant que les autres était ajouté à l'ensemble des ressources de l'expérimentation.

Pour l'évaluation des performances dans les différentes machines virtuelles. Chaque machine a une configuration spécifique. Toutes les machines exécutent le système d'exploitation Ubuntu 12.04 LTS. Les tests ont été menés sur un algorithme de détermination de position d'un satellite géostationnaire. L'algorithme utilise le filtre de Kalmen (EKF) pour obtenir la position visée et l'estimation de sa vitesse. L'auteur simule que l'application est demandée par une trentaine d'utilisateurs en concurrence sur l'algorithme de tracking EKF. Le temps pour finir chaque requête est la métrique de comparaison. Les résultats montrent que les bons résultats s'affichent sur la plus grande machine virtuelle. Par contre, l'auteur indique bien que l'augmentation de nombre de core (CPU) ne signifie pas forcément l'augmentation des performances.

Le programme spatial égyptien lui aussi a mis main dans ce sujet de recherche. Dans un article de (Abdelfattah El-Sharkawi et al, 2013), l'auteur présente une combinaison entre les services orientés objets et le Cloud Computing [193]. Pour le traitement de télémétrie avec un cas d'étude et une comparaison avec le projet Européen SSA. La motivation derrière ce travail revient á une volonté de programme d'investir dans le fournissement de service de traitement des télémétries des satellites de même série qui appartient à autres partenaires (Pays, petite Organisation, universités …etc.)(**Figure 67**). Le programme spatial égyptien avec un satellite d'observation de la terre et une seule station de sol au Caire n'aura qu'un contact de 10 minutes pour opérer son satellite. Avec la construction de plusieurs station de sol, le contact avec le satellite augmente, de ce fait. Le programme spatial égyptien trouve qu'ont suivant cette approche, il devient possible d'offrir un service de contact de satellite basé sur le modèle économique « payement á la consommation » pour les propriétaires de satellite de même série (modèle).

La solution proposée prend en considération les structures des paquets standards d'ECSS (EuropeanCooperation for SpaceStandardization) et utilise un modèle modulaire (Service Oriente Objet) pour le développement des missions et traitement des données des missions d'observation de la terre. Les étapes génériques d'exécution pour le traitement des données sont les suivantes :

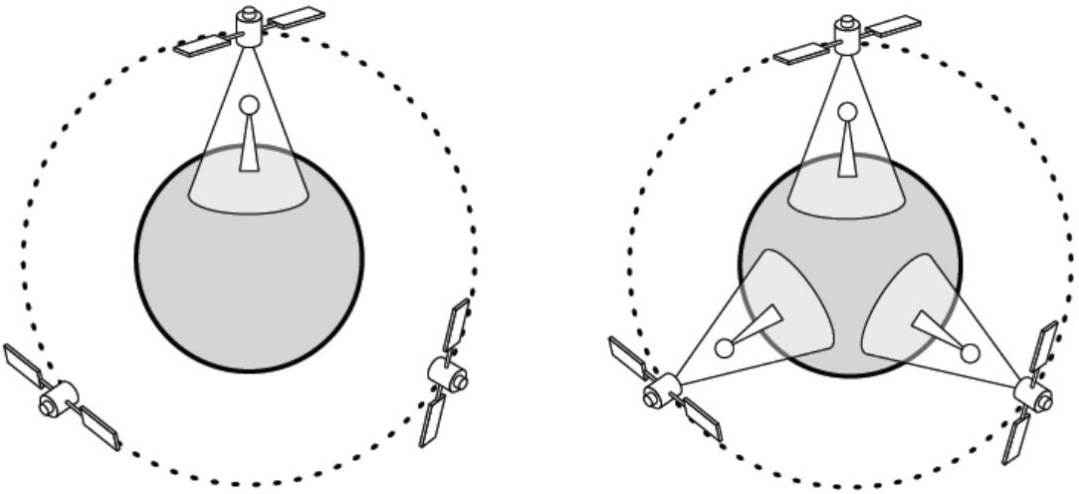

Figure 67: unique et multiple stations de sol

- Réception des données depuis le satellite
- Extraction des données des télémétries
- Extraction des valeurs brutes de chaque TM
- Application de routine de calibration des valeurs pour l'obtention des valeurs significatives
- Visualisation de la TM pour les clients connectés
- Archiver les données TM

En se basant sur ces étapes génériques la solution proposée est illustrée dans la **Figure 68**qui contient l'architecture modulaire ou les modules principaux sont :

- Système de journalisation
- Package de détermination d'orbite
- Base de données de télémétries
- Un moteur de traitement des données.

L'architecture était simulée et l'évaluation des performances réseaux sont fait sur le simulateur OPNET.

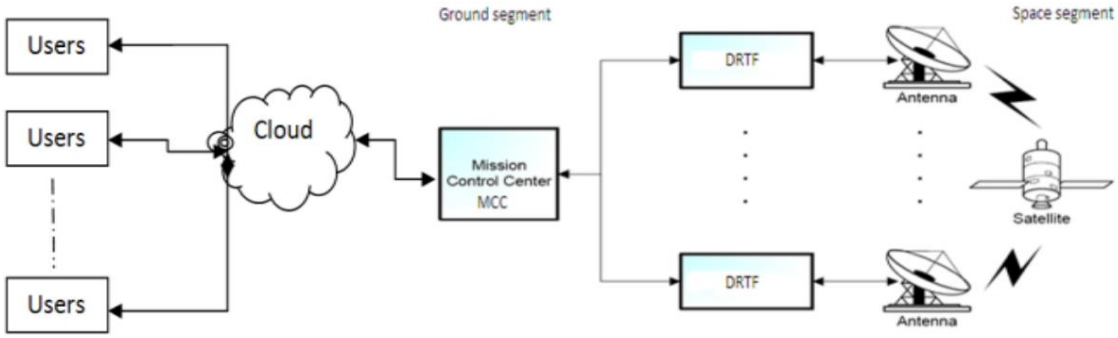

Figure 68 : Architecture proposée de PSE

5.4 Application de protocole proposé dans un scenario spatial :

Notre protocole permettra á la communauté Cloud de bénéficier d'une bonne relation sur laquelle ils peuvent aboutir à une amélioration de la communication inter-Cloud et permettre la collaboration entre eux. Dans notre travail et pour montrer comment notre protocole peut intégrer n'importe quel contexte Cloud, nous nous sommes intéressés sur le cout de conception missions spatiales. Avec un exemple, on peut voir comment le protocole et la collaboration inter-Cloud (inter-agences) augmente la fiabilité en terme de disponibilité de ressource et consommation. La plupart des missions spatiales ont pour but scientifique qui consomme beaucoup de stockage, communication, et ressources de calcul. Beaucoup plus, le mouvement et le dynamisme des missions en orbite imposent certain contrainte sur l'architecture réseau et le système de station de sol. Souvent, les missions spatiales nécessitent des stations de sol multiples dans différents lieux mondiaux. L'objectif d'avoir une multitude de stationsde sol dispatchées dans différents lieux et l'augmentation de temps de contact avec le satellite et d'avoir beaucoup plus de temps pour télécharger une grande quantité de donnée scientifique. Il est connu que les stations de sol sont trop chères, alors si on pense faire plusieurs, ceci augmentera de plus en plus le cout. D'autre part, les petites organisations utilisent leurs stations de sol pour un seul satellite pour 40 minutes par jour seulement alors que le reste de la journée la station est en repos. La solution est de considérer chaque station de sol comme étant un fournisseur de service Cloud avec une multitude de service qui peuvent être offerts comme le suivi, lien de communication de couverture, de stockage ou de calcul …etc. De cette façon, les petites agences et les grandes agences spatiales vont bénéficier de ce modèle. Les petites agences spatiales peuvent gagner de l'argent au lieu de garder leurs stations en repos durant la période de non fonctionnement. De l'autre côté, les grandes agences peuvent utiliser ces stations comme service á la demande. Rendre l'architecture classique de station basée sur le modèle Cloud Computing était une idée déjà sous étude qui renforce notre contribution et notre hypothèse d'utilisation de Cloud Computing dans les missions spatiales.

La conception de protocole prend en considération la capacité de supporter plusieurs types de services liés à des domaines plus spécifiques. Chaque type de service utilise des métriques pour déterminer la qualité de service. Cette capacité est garantie par les caractéristiques suivantes du protocole :

- La table de routage définit des champs obligatoires dont le Cloud ID, Service ID, Path, interface de sortie. Pour permettre á plusieurs métriques qui dépendent du type de service, la table de routage des services peut être développée (étendue) avec des nouveaux champs ajoutés à la fin des champs obligatoires définis par le standard de protocole. Ceci permettra à n'importe quel service spécifique à être traité par le protocole.
- La politique de processus de sélection ne dépend pas du protocole mais du fournisseur Cloud. Avec la possibilité d'étendre la table de routage de services, les fournisseurs Cloud devient capable de sélectionner le meilleur entré de la table de routage en se basant sur les métriques liées au service de domaine spécifique.
- Les structures des messages de protocole permettent à n'importe qu'elle métrique d'être formatée et sérialisée dans les messages grâce au pattern simple suivi pour représenter les paramètres, QoS et les valeurs des métriques (ID, longueur de champ, valeur).

- La décision d'acceptation est liée à une classe de métriques et paramètres de QoS. Le protocole prend en considération la façon dont les métriques soient traitées tout dépend de la classe des métriques. Les métriques qui dépendent seulement du nœud Cloud final sont les seules qui sont passées par les nœuds intermédiaires. Les métriques liées au lien et les nœuds intermédiaires ont un traitement spécifique comme l'accumulation des valeurs. La décision d'acceptation peut être prise dans la direction uplink et downlink. Le protocole prend en considération toutes les classes de métriques pour permettre au protocole d'être capable d'adapter plusieurs métriques et des métriques des services spécifiques durant la session de requête de service. Notre application du domaine spatiale est un exemple de démonstration de la capacité de protocole de s'opérer avec n'importe quel domaine.

Par rapport à l'ensemble des points expliqués dans ce chapitre sur le positionnement multiple des stations de sol à travers le monde. Pour rapprocher notre application de la réalité, nous avons créé une architecture réseaux de plusieurs nœuds. Chaque nœud correspond á une station de sol localisée dans un pays (Station Cloud). Les positions de différentes stations sont réelles c'est-à-dire que dans ces endroits il existe réellement une station sol opérationnel. La **Figure 69** représente la distribution des stations de sol dans les différents continents.

Figure 69: positions des stations de sol choisies dans l'expérimentation de protocole

Nous avons choisi des services représentatifs. Parmi les services il y'a le service de suivi (tracking), service de stockage, communication S-band, X-band, C-Band et K-band, service de correction radiométrique et géométrique. Nous avons attribué les services sur chaque station de sol. Le tableau 3 résume la relation station-service.

Tracking	Storage	S-Band	C-Band	X-Band	K-Band	Correction Radiométrique	Correction Géométrique
Tous GSs	GS1 GS4 GS8	All GSs	GS7 GS5	GS3 GS7 GS5	GS5 GS4	GS4	GS4

Tableau 3 : Association des services aux stations de sol

Une des taches à laquelle l'operateur s'intéresse beaucoup dans une mission spatiale est l'évitement de collision. Dans le cas où l'operateur n'a pas d'opportunité d'exécuter un bon plan pour changer l'orbite, le satellite peut avoir un grand risque d'être percuté par un autre

objet et fini la vie de la mission. Pour cette raison, un service est mis en disponibilité par les USA.

Dans ce scénario une alerte de collision est envoyée par une organisation spécialiste á la station GS1(USA). Un plan doit être planifié et envoyé au satellite. Mais, si l'équipe attend jusqu'au passage suivant au-dessus de la station GS1 alors ça sera trop tard. En utilisant notre solution, cette opération peut être automatisée pour collaborer avec une station de sol qui est dans un lieu favorable et conforme au meilleur temps de contact avec le satellite. Pour cette opération, une communication S-Band avec une fréquence spécifique est exigée.

Le processus de sélection sera basé sur le lieu de la station de sol, le plus court chemin, de plus, la capacité de system de station de sol d'offrir une communication S-band en uplink. Par contre, l'acceptation de service est basée sur la disponibilité de l'antenne de station dans le période demandé avec une bande passante choisie.

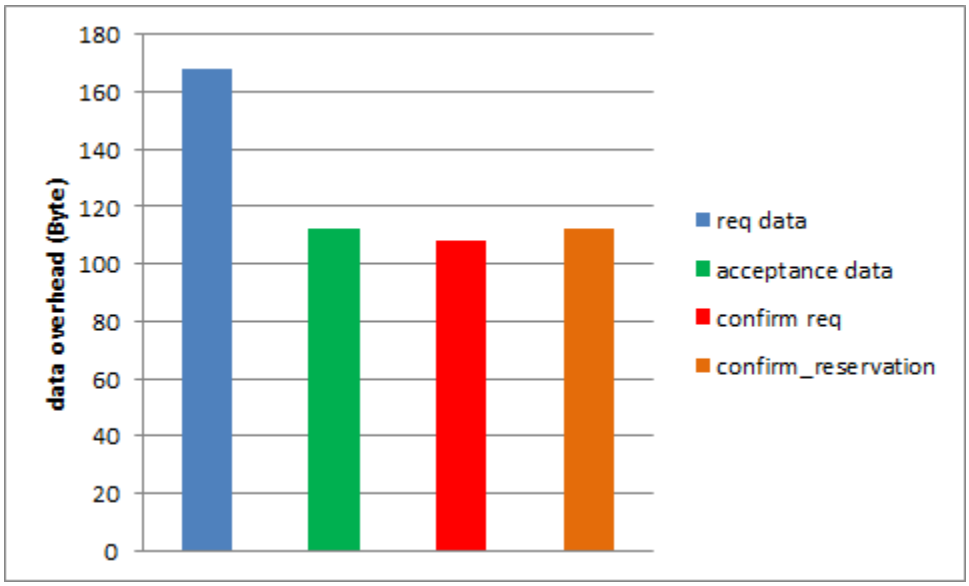

Figure 70 : taux de donnée lors d'une session de demande de service spatiale

La **Figure 70** est simple dans son contenu. Elle représente la quantité des données générées entre GS1 et GS8 pour demander un service de communication s-band. Les résultats montrent une différence trop petite entre les données de message de contrôle d'acceptation, message de confirmation et le message de réservation. Le rapprochement entre les valeurs est dû à l'utilisation de la même structure de message. Le message d'acceptation et de confirmation contient tous les deux un champ en plus de mise á jour des services et des paramètres de service. Les deux messages (acceptation et confirmation) atteignent 112 Bytes, alors que le message de confirmation de la requête atteint 108 Bytes. Le message de requête a un taux très élevé qui atteigne 168 Bytes car ce message contient plus d'information que n'importe qu'elle autre message de négociation de la session. Par contre, ceci est toujours considéré comme un taux trop faible comparé au liens établis entre les Clouds.

La quantité de données augmente linéairement en fonction de la distance c'est-à-dire le nombre intermédiaire entre la source et la destination. De plus, le taux de donnée augmente par rapport au nombre d'erreur qui peuvent apparaitre ou les requêtes non acceptées par les nœuds finaux ou intermédiaires. Dans ce cas, la quantité des données générées est petite

puisque le processus de session passe par une seule étape en générant juste un message d'erreur qui est de taille égale à 27 Bytes au minimum. L'établissement d'un nouveau lien est trop simple. Un message de notification d'erreur est généré et une nouvelle session sera ré-ouverte en suivant les mêmes étapes que la première session.

5.5 Conclusion

Le spatial est un domaine très coûteux, l'Algérie participe avec ses moyens dans l'exploitation d'espace. L'interopérabilité entre les agences et organisations spatiales est un chemin inévitable. Les grandes comme les petites organisations vont bénéficier de ce modèle économique. La présentation des stations de sol sous forme de service á la demande va permettre d'optimiser l'utilisation des ressources de sol. Si on considère les stations comme des fournisseurs Cloud, nous devons définir un moyen de communication. Notre proposition de protocole de communication dans un environnement Cloud peut être utilisée dans le cas des stations de sol des satellites. L'application spatiale est une application réaliste et notre protocole a montré comment il s'est mis á communiqué les services dédiés aux sciences de l'espace. Ceci montre aussi que le protocole est applicable dans de multi-domaines.

6.1 Conclusion générale :

Le Monde IT parle souvent de technologie Cloud. L'ensemble des organisations doivent se préparer pour s'intégrer avec le Cloud. Nous remarquons une nouvelle méthode pour l'offre des services informatique. Nous voyons nos boites d'émail se développer pour devenir beaucoup plus qu'un outil d'échange d'émail et des fichiers. Nous pouvons maintenant stocker, faire du traitement de texte, partager des documents. Nous pouvons se connecter à internet pour exploiter un grand nombre de ressource depuis notre machine distante qui ne contient rien d'une connexion internet et un explorateur web. Le monde avance, et on doit avancer avec. Nous avons participé dans un des problème de Cloud qui est au niveau abstrait plus haut l'interopérabilité alors qu'au niveau plus bas c'est la garantie de niveau de service de bout en bout entres fournisseurs Cloud. Nous avons participé avec la création d'un protocole généralisé qui englobe les opérations d'échange de l'information concernant les services proposés sur chaque Cloud, la négociation des services avec exigence de qualité de service, la réservation et la libération des ressources. La conception des structures des messages et de la table de routage permet à différent Cloud offrant des services spécialisés. La sécurité est un autre problème qui s'impose en parallèle avec l'interopérabilité. Nous avons laissé un point de raccord avec notre protocole qui permettra la possibilité d'intégrer le concept de sécurité dans le nouveau protocole. Sachant que l'Algérie est un pays qui s'engage dans le domaine spatial. Nous avons pris l'exemple d'étudier un cas d'utilisateur pour l'utilisation de Cloud Computing dans les stations de sol et la façon dont ils peuvent communiquer pour augmenter la disponibilité. Avec un tel exemple, nous avons fait une démonstration sur la façon par quelle le protocole peut etre utilisé dans un domaine particulier.

Références:

[1] A.Roussignol (2014), 25 Definitions of Cloud Computing. [Online]: https://www.linkedin.com/pulse/20141117105234-958990-25-definitions-of-cloud-computing (accédé le 16/03/2016)

[2] National Institute of Standards and Technology (2009), About NIST. [Online]: http://www.nist.gov/public_affairs/nandyou.cfm (accédé le 16/03/2016)

[3] R.Buyya et al (2008), Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing UtilitiesProceeding. HPCC '08 Proceedings of the 2008 10th IEEE International Conference on High Performance Computing and Communications Pages 5-13. doi>10.1109/HPCC.2008.172

[4] Google App Engine (Online): https://cloud.google.com/AppEngine (accédé le 16/03/2016)

[5] Micosoft Azure (Online): https://azure.microsoft.com/en-gb/overview/what-is-azure/ (accédé le 16/03/2016)

[6] A Border Gateway Protocol 4 (BGP-4) (2006), [Online]: https://tools.ietf.org/html/rfc4271 (accédé le 16/03/2016)

[7] Xenproject (2013) [Online] http://www.xenproject.org/project-mission.html (accédé le 16/03/2016)

[8] KVM,OpenVirtualization Alliance. [Online]: https://openvirtualizationalliance.org/what-kvm/overview (accédé le 17/03/2016)

[9] VMware, [Online] http://www.vmware.com/ (accédé le 17/03/2016)

[10] Geni, [Online] http://www.geni.net/ (accédé le 17/03/2016)

[11] P.D Patet (2014), Live Virtual Machine Migration Techniques in Cloud Computing: A Survey. International Journal of Computer Applications

[12] D.M Barringtom, (2006) CMPSCI 311 Discussion #11: Bin Packing. [Online]: https://people.cs.umass.edu/~barring/cs311/disc/11.html (accédé le 17/03/2016)

[13] K.SRao (2015), Heuristics based server consolidation with residual resource defragmentation in cloud data centers, Future Generation Computer Systems. doi:10.1016/j.future.2014.09.009

[14] Virtual Machine Consolidation in Cloud Data Centers Using ACO Metaheuristic, Euro-20th International Conference, Porto, Portugal, August 25-29, 2014 Proceedings. DOI 10.1007/978-3-319-09873-9_26

[15] M. Marzolla (2011) Server consolidation in Clouds through gossiping.World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium.10.1109/WoWMoM.2011.5986483

[16] SPECvirt_sc2010 (2013). [Online]: https://www.spec.org/virt_sc2010/ (accédé le 17/03/2016)

[17] Open Web Application Security Project (2015), [Online]: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project. (accédé le 17/03/2016)

[18] Bezemer C-P et al (2010): Multi-tenant SaaS applications: maintenance dream or nightmare? In Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE), Antwerp, Belgium. NY, USA: ACM New York; 2010:88–92. doi>10.1145/1862372.1862393

[19] Cloud Security Alliance (2012), Security guidance for critical areas of Mobile Computing. [Online]: https://downloads.cloudsecurityalliance.org/initiatives/mobile/Mobile_Guidance_v1.pdf (accédé le 17/03/2016).

[20]Xu K, et al (2009). Mobile Mashup: Architecture, Challenges and Suggestions. In International Conference on Management and Service Science.MASS'09. Washington, DC, USA: IEEE Computer Society; 2009:1–4. Doi>10.1109/ICMSS.2009.5301595

[222] W Dawoud et al (2010). Infrastructure as a service security: Challenges and solutions. In the 7th International Conference on Informatics and Systems (INFOS), Potsdam, Germany. Washington, DC, USA: IEEE Computer Society; 2010:1–8.

[22] JS Reuben (2007): A survey on virtual machine Security. Seminar on Network Security; Helsinki University of Technology.[Online]: http://www.tml.tkk.fi/Publications/C/25/papers/Reuben_final.pdf (accédé le 17/03/2016)

[23] IEEE, Standards Glossary. [Online]: https://www.ieee.org/education_careers/education/standards/standards_glossary.html. (accédé le 17/03/2016).

[24] Cloud Data Management Interfaces (CDMI). [Online] : http://www.snia.org/cdmi (accédé le 17/03/2016).

[25] Open Cloud Interoperability Framwork. [Online]: http://www.ocean-project.eu/bin/view/Services/Open_Cloud_Interoperability_Framework. (accédé le 17/03/2016).

[26] The Linux Information Project (2005), Best Effort Definition. [Online]: http://www.linfo.org/best_effort.html (accédé le 23/03/2016)

[27] Cisco IOS Quality of Service Solutions Configuration Guide, Quality of Service Overview [Online]: http://www.cisco.com/c/en/us/td/docs/ios/12_2/qos/configuration/guide/fqos_c/qcfintro.pdf (accédé le 23/03/2016)

[28] Information Sciences Institute, University of Southern California (1981) INTERNET PROTOCOL. [Online] http://www.ietf.org/rfc/rfc0791.txt (accédé le 23/03/2016)

[29] J. Poste (1981), SERVICE MAPPINGS. [Online]: https://tools.ietf.org/html/rfc795 (accédé le 23/03/2016).

[30] S. Blake et al (1998), Network Working Group, An Architecture for Differentiated Services. [Online]: https://tools.ietf.org/html/rfc2475#section-2 (accédé le 23/03/2016)

[31]YoramBernet et al (1999), A Framework for Differentiated Services, Internet Draft. [Online]: https://www.ietf.org/proceedings/44/I-D/draft-ietf-diffserv-framework-02.txt (accédé le 24/03/2016)

[233] Cisco, DiffServ -- The Scalable End-to-End QoS Model.[Online] : http://www.cisco.com/en/US/technologies/tk543/tk766/technologies_white_paper09186a00800a3e2f_ps6610_Products_White_Paper.html (accédé le 24/03/2016).

[33] C. Hedrick (1988), Routing Information Protocol, Network Working Group. [Online]: https://www.ietf.org/rfc/rfc1058.txt (accédé le 24/03/2016)

[34] J. Moy (1998), OSPF version 2, Network Working Group. [Online]: http://tools.ietf.org/html/rfc2328. (accédé le 26/03/2016)

[35] D. Oran (1990). OSI IS-IS Intra-domain Routing Protocol. Network Working Group. [Online]: http://tools.ietf.org/html/rfc1142. (accédé le 26/03/2016)

[36]juniper (2012), Understanding OSPF Traffic Control. [Online]: http://www.juniper.net/documentation/en_US/junos12.3/topics/concept/ospf-traffic-control-overview.html. (accédé le 26/03/2016).

[37] C. Villamizar and T. Li (1998), "IS-IS Optimized Multipath (IS-IS OMP)", Internet draft. [Online]: https://tools.ietf.org/html/draft-ietf-isis-omp-01 (accédé le 26/03/2016).

[38] G. Apostolopoulos et al (1999). QoS Routing Mechanisms and OSPF Extensions, Network Working Group. [Online]: https://tools.ietf.org/html/rfc2676 (accédé le 21/04/2016)

[39]Roch A. Gu´erin et al (1997) QoS Routing Mechanisms and OSPF Extensions, Global Telecommunications Conference, 1997. GLOBECOM '97. IEEE (Volume:3). DOI: 10.1109/GLOCOM.1997.644603

[40] Y. Goto, et al (1997), "Path QoS Collection for Stable Hop-by-hop QoS Routing", Proceedings of INET '97. [Online]: https://www.isoc.org/inet97/proceedings/F4/F4_2.HTM (accédé le 21/04/2016)

[41] Zheng Wang (1996), Quality-of-service routing for supporting multimedia applications, IEEE Journal on Selected Areas in Communications (Volume:14 , Issue: 7). DOI: 0.1109/49.536364

[42] E. Rosen (2001), Multiprotocol Label Switching Architecture, Network Working Group. [Online]: https://tools.ietf.org/html/rfc3031 (accédé le 21/04/2016).

[43] L. Andersson (2007) Experience with the Label Distribution Protocol (LDP), Network Working Group. [Online]: https://tools.ietf.org/html/rfc5037 (accédé le 21/04/2016)

[44] D. Awduche (2001), RSVP-TE: Extensions to RSVP for LSP Tunnels, Network Working Group. [Online]: https://tools.ietf.org/html/rfc3209 (accédé le 21/04/2016).

[45] Marcel Waldvogel (1997). Scalable high speed IP routing lookups.SIGCOMM '97 Proceedings of the ACM SIGCOMM '97 conference on Applications, technologies, architectures, and protocols for computer communication.Pages 25-36.doi>10.1145/263105.263136.

[46] S. Nilson (1998). Fast address look-up for internet routers. BC '98 Proceedings of the IFIP TC6/WG6.2 Fourth International Conference on Broadband Communications: The future of telecommunications.

[47] Eric C. Rosen (2004), BGP/MPLS IP VPNs. Network Working Group . [Online]: https://tools.ietf.org/html/draft-ietf-l3vpn-rfc2547bis-03. (accédé le 21/04/2016)

[48] R. Aggarwal (2007), Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs). [Online]: https://tools.ietf.org/html/rfc4875 (accédé le 21/04/2016)

[49] j. Cucchiara (2004), Definitions of Managed Objects for the Multiprotocol Label Switching (MPLS), Label Distribution Protocol (LDP). [Online]: https://www.ietf.org/rfc/rfc3815.txt (accédé le 21/04/2016)

[50] J. Ash (2002). Applicability Statement for CR-LDP, Network Working Group.[Online]: https://tools.ietf.org/html/rfc3213 (accédé le 21/04/2016)

[51] E. Crawley (1998). A Framework for QoS-based Routing in the Internet. Network Working Group. [Online]: https://tools.ietf.org/html/rfc2386 (accédé le 21/04/2016)

[52] R. Braden (1994). Integrated Services in the Internet Architecture: an Overview. Network Working Group. [Online]: https://tools.ietf.org/html/rfc1633 (accédé le 21/04/2016)

[53] S. Shenker (1997).Specification of Guaranteed Quality of Service. Network Working Group. [Online]: https://tools.ietf.org/html/rfc2212 (accédé le 21/04/2016)

[54] R. Braden (1997). Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification. [Online]: https://tools.ietf.org/html/rfc2205. (Accédé le 21/04/2016)

[55]D. Ferrari (2002). A scheme for real-time channel establishment in wide-area networks.IEEE Journal on Selected Areas in Communications (Volume:8 , Issue: 3). DOI: 10.1109/49.53013

[56]ruce A Mah (1993). Technical Report: A Mechanism for the Administration of Real-Time Channels (ACM).

[57] Y. Rekhter (1995) A Border Gateway Protocol 4 (BGP-4). Network Working Group. [Online]: https://www.rfc-editor.org/rfc/rfc1771.txt (Accédé le 21/04/2016).

[58]Iljitsch Van Beijnum (2002). Building a reliable networks with the Border Gateway Protocol. BGP.O'REILLY Editor.

[59] Eric C. Rosen (1982) EXTERIOR GATEWAY PROTOCOL (EGP. [Online]: https://tools.ietf.org/html/rfc827 (Accédé le 21/04/2016)

[60]www.iana.org

[61] G. Cristallo (2004). The BGP QOS_NLRI Attribute. Network Working Group. [Online]: https://tools.ietf.org/html/draft-jacquenet-bgp-qos-00 (Accédé le 21/04/2016).

[62]Florin Pop (2008). Communication Model for Decentralized Meta-Scheduler in Grid Environments.Complex, Intelligent and Software Intensive Systems, 2008.CISIS 2008.International Conference on. DOI: 10.1109/CISIS.2008.131

[63]Mihaela-AndreeaVasile et Al (2015). Resource-aware hybrid scheduling algorithm in heterogeneous distributed computing.Future Generation Computer Systems. Volume 51, October 2015, Pages 61–71. DOI : 10.1016/j.future.2014.11.019.

[64] Andrei Sfrent et al (2015). Asymptotic scheduling for many task computing in Big Data platforms.Information Sciences. doi:10.1016/j.ins.2015.03.053

[66] L. Mohammad Khanli et al () Grid-JQA: Grid Java based Quality of service management by Active database. Conferences in Research and Practice in Information Technology.

[67] V anishTalwar (2004) Architecture for resource allocation services supporting interactive remote desktop sessions in utility grids. MGC '04 Proceedings of the 2nd workshop on Middleware for grid computing.Pages 23-28. Doi:10.1145/1028493.1028497

[68]Sushant Sharma (2011), End-to-end network QoS via scheduling of flexible resource reservation requests. 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC). DOI: 10.1145/2063384.2063475

[69]YasserMansouri et al (2008). Optimal Number of Replicas with QoS Assurance in Data Grid Environment.AMS '08 Proceedings of the 2008 Second Asia International Conference on Modelling & Simulation (AMS).Pages 168-173. DOI: 10.1109/AMS.2008.147

[70]BakhtaMeroufel (2013), Managing Data Replication and Placement based on Availability. AASRI Procedia.doi:10.1016/j.aasri.2013.10.071

[71] Oscar Encina C et Al (2014), Proceedings of the 8th Nordic Conference on Pattern Languages of Programs. doi>10.1145/2676680.2676688

[72] AdelNadjaranToosi (2014), Interconnected Cloud Computing Environments: Challenges, Taxonomy, and Survey. Journal ACM Computing Surveys (CSUR). Volume 47 Issue 1, July 2014. Article No. 7. Doi:10.1145/2593512

[73]Jingxin K. Wang (2012), Interoperability and Standardization of Intercloud Cloud Computing. [Online]: http://arxiv.org/pdf/1212.5956.pdf (Accédé le 21/04/2016).

[74] P. Saint-Andre (2011). Extensible Messaging and Presence Protocol XMPP):Instant Messaging and Presence. Internet Engineering Task Force (IETF).[Online]: https://tools.ietf.org/html/rfc6121 (Accédé le 21/04/2016).

[75] G. Kecskemeti (2012), Facilitating Self-Adaptable Inter-cloud Management.2012 20th Euromicro International Conference on Parallel, Distributed and Network-based Processing. DOI: 10.1109/PDP.2012.41

[76] Mohammad Mehedi Hassan (2010), Horizontal Dynamic Cloud Collaboration Platform: Research Opportunities and Challenges. [Online]: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.232.4468&rep=rep1&type=pdf (Accédé le 21/04/2016)

[77] P. Radha Krishna Et al (2005), From Contracts to E-Contracts: Modeling and Enactment. Information Technology and Management, Volume 6, Issue 4, pp 363-387. DOI: 10.1007/s10799-005-3901-z

[78] Dhigetoshi Yokoyama (2014), Network traffic optimization architecture for scalability in academic inter-cloud computing environments. Proceedings of the 2nd International Workshop on Hot Topics in Cloud service Scalability. Doi:10.1145/2649563.2649572

[79] WalterCerroni (2014), Performance of network and computing resource sharing in federated cloud systems. DCC '14 Proceedings of the 2014 ACM SIGCOMM workshop on Distributed cloud computing. Pages 25-30. DOI: 1 0.1145/2627566.2627567

[80]: Muhammad Bilal Amin (2012), Intercloud message exchange middleware. Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication. Article No. 79. Doi:10.1145/2184751.2184845

[81]: Yonggang Wen et al (2011), Designing an inter-cloud messaging protocol for content distribution as a service (CoDaaS) over future internet. Proceedings of the 6th International Conference on Future Internet Technologies. DOI: 10.1145/2002396.2002420

[82]: P. Resnick (2001), Internet Message Format. Network Working Groupe.[Online]: https://www.ietf.org/rfc/rfc2822.txt (Accédé le 21/04/2016)

[83]: (2007), Web Services Description Language (WSDL) Version 2.0 Part 1. [Online}: https://www.w3.org/TR/wsdl (Accédé le 21/04/2016)

[84]: (2000), Simple Object Access Protocol (SOAP) 1.1. [Online]: https://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[85]: WassimItani et al (2014) ServBGP: BGP-inspired autonomic service routing for multi-provider collaborative architectures in the cloud. Future Generation Computer System. DOI: 10.1016/j.future.2012.05.013

[86] NASA JPL (2010) Mars Rovers Mission Using Cloud Computing. [Online]: http://www.jpl.nasa.gov/news/news.php?feature=2799 (accédé le 27/05/2016)

[87]Ali Llewellyn (2012). Nebula, NASA, and OpenStack.[Online]: https://open.nasa.gov/blog/nebula-nasa-and-openstack/ (accédé le 27/05/2016)

[88] MehranSarkarati et al (2012). When the Space Gets Cloudy A Systematic Assessment of Cloud Computing Application Domains for ESA Ground Data Systems.[Online]: http://spaceops2012.org/proceedings/documents/id1289207-Paper-001.pdf (accédé le 27/05/2016)

[89] Mehran Sarkarati1 et al (2014), Mission Operations as a Service Cloud Computing for Space Missions beyond Infrastructure-as-a-Service.SpaceOps 2014 Conference. DOI: 10.2514/6.2014-1776

[90] Steven J Johnston et al (2013), Clouds in Space: Scientific Computing using Windows Azure. Journal of Cloud Computing. DOI: 10.1186/2192-113X-2-2

[91] BingweiLiua et al (2013), Cloud-based Space Situational Awareness: Initial Design and Evaluation. [Online] : http://harvey.binghamton.edu/~bliu11/cv/publications/SPIE-DSS13.pdf (Accédé le 28/05/2016).

[92] Abdelfattah El-Sharkawi (2013). Service Oriented Architecture for Remote Sensing Satellite Telemetry Data Implemented on Cloud Computing. I.J. Information Technology and Computer Science. DOI: 10.5815/ijitcs.2013.07.02

Figures

Tableaux

Acronymes :

SOA: Service Oriented Architecture

IT: Information Technology

SLA: Service Level Agreement

BGP: Border Gateway Protocol

NIST National Institude of Standards and Technology:

LAN: Local Area Network

TCPIP: Transmission Control Protocol Internet IP

KVM: Kernal-base Virtual Machine

API: Application Programming Interface

OSI: Open System Interconnection model

ERP: Enterprise Resources Planning

CRM: Customer Relationship Management

OWASP: Open Web Application Security Project

CPU: Central Processing Unit

IEEE: Institute of Electrical and Electronics Engineers

FTP: File Transfer Protocol

CDMI: Cloud Data Management Interface standard

MPLS: Multiprotocol Label Switching

RSVP: Resource Reservation Protocol

TOS: Type of Service

DS: Differentiated services

ISP: Internet Service Provider

BA: Behaviour Aggregate

MF: Multi Field

SLS: Security Level System

RIP: Routing Information Protocol

OSPF: Open Shortest Path First

ISIS: intermediate system to intermediate system

ECMP:Equal Cost Multi Path

TE: Traffic Engineering

BF: Bellman Ford

MPLS: Multi-Protocol Layer Switching

COS: Class Of Service

TTL: Time To Live

LSR: Label Switch Router

LSP: Label Switch Path

VC: Virtual Circuit

ER: Explicit Route

FEC: Forward Error Correction

EGP: Exterior Gateway Protocol

IGP: Interior Gateway Protocol

IANA: International Assigned Numbers Authority

AS autonomous system

RIR: Regional Internet Registry

NLRI: Network Layer Reachability Information

QOS: Quality Of Service

FAI: Fournisseur d'accès Internet

ONR: Optimal Number Of Replication

XMPP: Extensible Messaging and PresensePtococol

UDF: Uniform Data Format

KM: Knowledge Management

DCC: Dynamic Clouds Collaboration

AIC: Academic inter-Cloud Architecture

DDS: Data Distribution Service

OWL: Ontology Web Language

SIP: Session Initiation Protocol

SMTP: Simple Mail Transfer Protocol

XML: Extensible Markup Language

TLV: Text Length Value

RPC: Remote Procedure Call

SOAP: Simple Object Access Protocol

WSDL: Web Service Description Language

CSP: Cloud Service Provider

SAM: Service Advertisement Message

MTU: Maximum Transmission Unit

NASA: National Aeronautics and Space Administration

ESA: European Space Agency

ISS: Internation Space Station

GPS: Global Positioning System

JPL : Jet Propublsion Lab

MS: Microsoft

ESRIN: European Space Research institute

ESOC: European Space Operations Center

SSA: Space Situational Awareness

RF: Radio Frequence

EKF: Extended Kalmen Filter