Les Propositions

Après avoir analysé le contexte, et rapporté les principaux résultats de l'existant et du background, nous allons consigner dans ce chapitre nos propositions qui visent à offrir à l'utilisateur des services personnalisés, avec une continuité de session et une QoS de bout en bout, et cela dans un contexte qui est totalement hétérogène et mobile. Nos contributions sont d'ordre organisationnel, fonctionnel et protocolaire. Elles se subdivisent en trois principales propositions. La première est relative à la dimension organisationnelle (§V.1), qui adresse notre problématique d'avoir une architecture décentralisée avec le maximum de flexibilité et de dynamicité, afin de garantir une continuité de service dans un environnement hétérogène et avec des changements dynamiques. La seconde proposition est relative au composant de service autonomique (§V.2), dans cette partie nous repensons la notion du « composant de service » et nous proposons un composant de service qui soit auto-gérable et auto-adaptable grâce à un agent-QoS, pour faciliter le remplacement d'un composant de service par un autre composant ubiquitaire sans impacter la QoS du service global. La dernière proposition est la dimension protocolaire (§V.3), dans cette partie nous nous sommes intéressés aux besoins d'une session de service guidée par les préférences de l'utilisateur. Nous proposons un modèle protocolaire pour assurer une interaction plus flexible entre les différents acteurs de l'architecture, et cela pour faciliter la création et la modification d'une session de service unique et continue.

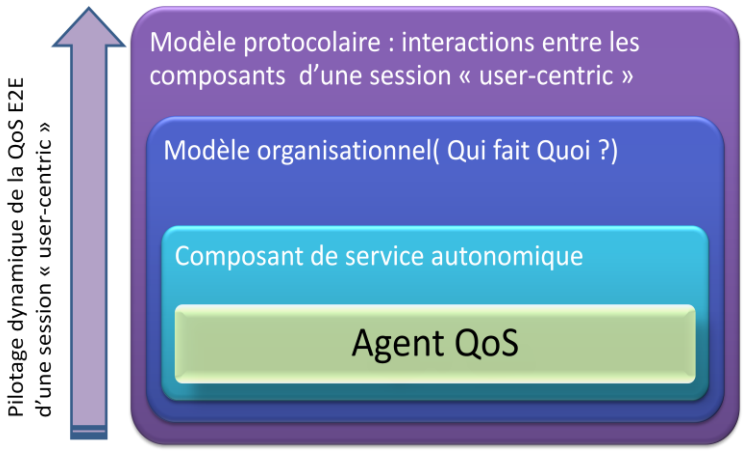

Figure 14: Propositions

V.1 La Dimension organisationnelle

Nous voulions, à travers ce chapitre, faire valoir l'idée que l'identification des acteurs et la séparation des rôles constitue une première solution efficace aux problématiques des architectures d'aujourd'hui, à savoir, la dynamicité, la réactivité, la flexibilité et la QoS de bout en bout. Donc, une nouvelle organisation permet de mieux piloter dynamiquement la qualité de service dans un nouvel environnement de l'utilisateur qui est totalement hétérogène et mobile. Ce chapitre est essentiellement consacré au développement d'un modèle organisationnel qui répond à la question « qui fait quoi ?» dans une architecture de volume et complexité importante. Ce modèle est bâti sur l'architecture UBIS développée par notre groupe de travail qui est structurée en niveaux de visibilité, qui sont des niveaux d'abstraction permettant de différencier les services rendus.

Notre proposition est la répartition des responsabilités de la QoS selon ses niveaux de visibilité de l'architecture UBIS pour faire apparaître les éléments architecturaux impactant la qualité de service de bout en bout:

- Le niveau responsable du service rendu par les équipements physiques est le VPEN, il gère la QoS des ressources physiques tels que la mémoire et la CPU pour garantir un computing delivery.

- Le niveau responsable du service rendu par le réseau de transport est le VPCN, il gère la QOS des couches 1, 2, 3 du modèle OSI responsables de la réalisation des fonctions de transport. Ce niveau est responsable du media delivery, il garantit la QoS du réseau de transport d'un flux média.
- Le niveau responsable de service rendu par le service applicatif est le VPSN, il gère la QoS du composant de service liée au traitement de l'information pour assurer un service Delivery.
- Et le niveau responsable du contrat de service établi entre les utilisateurs et les fournisseurs de service est le VPUN, il gère la QoS demandée par l'utilisateur représenté par son SLA (Service Level Agreement), et il gère aussi ses préférences qui peuvent changer au cours de la session.

Notre préoccupation est de définir une organisation capable de piloter dynamiquement la QoS de bout en bout à travers ces différents niveaux de visibilité. Dans la section suivante, nous allons présenter le modèle organisationnel de l'architecture UBIS (§V.1.1) pour répondre au besoin de l'utilisateur dans ce nouveau contexte qui est totalement hétérogène et mobile. Ce modèle se base sur l'association de deux concepts clés: les composants de service qui représentent les acteurs (§ V.1.2), et les rôles que joue l'agent QoS (§V.1.3) par rapport à la prise de décision lors des différents changements qui se produisent au cours de la session de l'utilisateur, et qui peuvent être liés à la mobilité ou bien à un dysfonctionnement d'un service.

V.1.1 Modèle organisationnel de l'architecture UBIS: Qui fait Quoi?

Dans l'architecture UBIS tout est considéré comme un service. A chaque niveau de visibilité de l'architecture, on peut trouver des composants de service qui rendent un service de type contrôle «CSC», Applicatif «ASC» ou gestion «GSC» qui participent aux différentes phases de la session «user-centric».

Dans une architecture de volume et complexité importantes, il est intéressant d'avoir multiples points de contrôle et de gestion qui participent au maintien du service avec le niveau de la QoS requis au cours de la session de l'utilisateur, et cela suivant une approche Top-

Down du niveau service (VPSN) au niveau utilisateur (VPUN) pour assurer un pilotage dynamique de la qualité de service de bout en bout.

Cet aspect organisationnel nous conduit à prendre comme notion clé l'agent-QoS qui sera intégré dans tout type de services de l'architecture à tous les niveaux de visibilité, c'est-à-dire que chaque composant de chaque niveau architectural sera piloté et invoqué en fonction de son comportement (QoS), qui est contrôlé et géré par son agent de QoS au cours de la session UBIS.

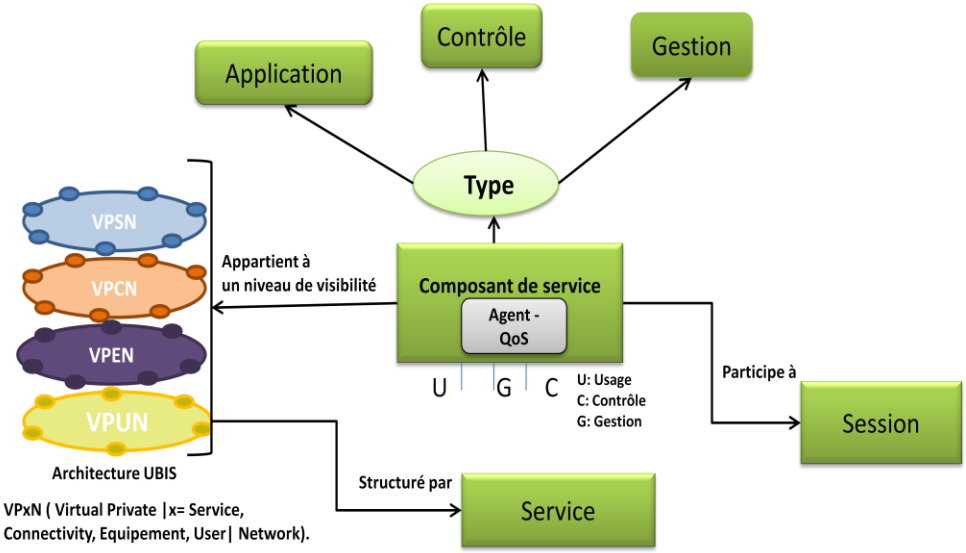

Figure 15: Modèle Organisationnel

.

V.1.2 Acteurs (Composant de service)

Le modèle organisationnel de l'architecture repose sur l'attribution des acteurs et cela pour améliorer la capacité d'actions lorsqu'un changement de QoS apparait suite à la mobilité (utilisateur, terminal, service, session) ou à un dysfonctionnement d'un composant de service. Pour ce faire, chaque acteur doit jouer un rôle organisationnel au sein du niveau de visibilité auquel il appartient (cross-layer) ou bien entre les niveaux de visibilité (intra-layer) pendant la phase de l'exploitation pour participer au processus d'adaptation et du maintient de la QoS de bout en bout pour l'utilisateur final, et il peut être:

- *Initiateur*: représente l'entité qui détecte le changement de QoS grâce à son agent de QoS au cours de la session d'un utilisateur, et se charge d'informer le décideur qui puisse se trouver soit dans le même niveau de visibilité N ou bien dans un niveau de visibilité supérieur (N+1) ou inferieur (N-1), pour qu'il prenne les décisions nécessaires pour maintenir la chaine de QoS de bout en bout.
- *Décideur*: représente l'entité qui se charge de prendre les mesures nécessaires pour alimenter les informations décisionnelles, conformément à ses responsabilités et cela en fonction de son niveau de visibilité.
- *Exécuteur*: représente l'entité qui contrôle le changement de la QoS et effectue les adaptations nécessaires.

V.1.3 Les rôles de l'agent de QoS

L'agent de QoS proposé est un élément générique qui offre une autonomie de traitement de la qualité de service à un composant de service pour avoir une meilleure performance de la QoS au cours de la session de l'utilisateur. Il prend en charge le processus de mesure des ressources internes et également le processus de communication, il assure une coopération au niveau horizontal entre les composants de service quand il s'agit des composants du même niveau de visibilité, et une coordination pour assurer l'agrégation des besoins QoS entre les différents niveaux de visibilité.

L'agent de QoS est une entité stable et selon la stratégie de l'organisation et les besoins au cours de la session des rôles lui sont attachés et qui peuvent être de type : Passif, Actif, Interactif ou Proactif.

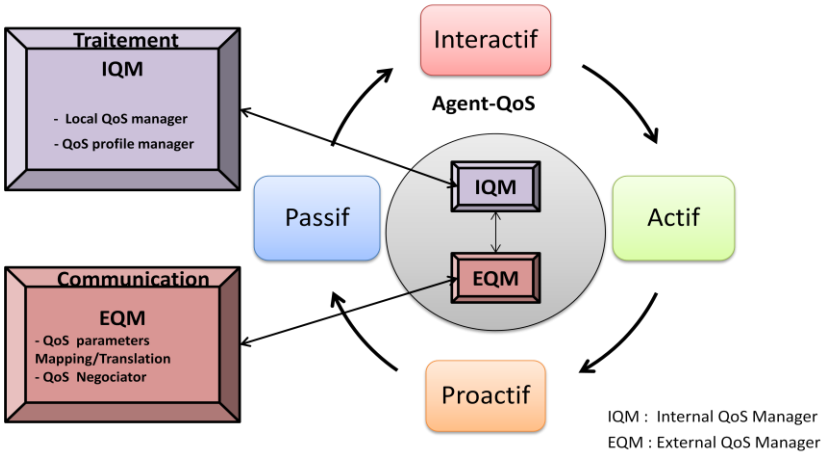

Figure 16: Rôles et fonctionnalités de l'agent QoS

- Le *rôle passif* désigne l'état où l'agent QoS assure uniquement le traitement interne; il mesure la QoS et met à jour les valeurs de QoS courantes. Dans le cas de non-respect du contrat de QoS c.-à-d. le dépassement des valeurs seuils, il ne communique pas son état à son environnement sauf dans le cas où il est sollicité par sa communauté pour l'informer dans quel état il se trouve.
- Le *rôle Actif* désigne l'état où l'agent-QoS joue le rôle de métrologue et de controleur de la QoS, et il notifie régulièrement, à qui a le droit, le statut de QoS de son composant de service, c'est-à-dire s'il respecte toujours son contrat de service ou bien s'il est hors de contrat de service (In contrat/Out contrat).
- Le *rôle Interactif* désigne l'état où l'agent-QoS est doté d'une capacité d'interactions avec d'autres Agent-QoS. Il peut négocier les paramètres QoS à maintenir dans un même niveau de visibilité, ou bien entre les composants de service à activer dans les autres niveaux de visibilité pour assurer le mappage entre les différents paramètres de QoS et par conséquent avoir une solution de bout en bout.
- Le *rôle Proactif* désigne l'état où l'agent QoS possède les connaissances et les règles qui lui permettent de prendre les décisions tout seul, pour remedier à un problème qui lui est propre ou qui relève de son domaine de responsabilité, et il doit envoyer des notifications à qui de droit.

V.1.4 Les fonctionnalités de l'agent QoS

Comme nous l'avons déjà présenté précédemment, L'agent QoS est une entité autonome intégrée dans chaque composant de service qui mesure la QoS en temps réel pour détecter la défaillance au bon moment durant les changements (mobilité, préférences, dégradation). Nous expliquons dans ce qui suit, les éléments fonctionnels de l'agent de QoS qui sont, d'une part, liés au traitement interne pour assurer une réservation des ressources selon la demande de l'utilisateur, et d'autre part, liés à la communication pour assurer une coordination et une coopération entre les différents composants de services appartenant aux différents niveaux de visibilité de l'architecture UBIS. L'agent QoS s'appuie sur deux éléments fonctionnels de gestion: IQM (Internal QoS manager) et EQM (External QoS manager)(Figure 16):

L'IQM est en charge de la gestion et le contrôle local de la qualité de service spécifique pour chaque composant de service. Il surveille les ressources internes du composant de service pour maintenir la QoS. Les fonctions principales d'IQM proposées sont les suivantes:

- *Local QoS Manager* : informe le plan de contrôle des ressources dont il a besoin pour assurer son traitement interne.
- *QoS Profile Manager* : gère le profil de la qualité de service pour chaque élément de service. Il assure en temps réel le traitement nécessaire pour connaître le statut du composant de service en termes de contrat de QoS pendant l'exploitation (In contrat/Out contrat). En effet, l'IQM s'interface avec les bases de données utilisateur et opérateur nommées respectivement Infosphère et Infoware pour l'échange d'informations nécessaires au traitement interne.

La principale fonction de l'EQM est la communication et la coordination entre les ressources QoS des différents niveaux de visibilité et il permet également le mappage des paramètres QoS entre les différents nœuds de l'architecture, ce qui permet une continuité de session en cas de dégradation de la qualité de service et ce quel que soit le type de mobilité. Les fonctions principales d'EQM proposées sont les suivantes :

- QoS Mapping/Translation Parameters : assure le mappage et la translation des mécanismes et paramètres de QoS utilisés. Ce qui permet la déclinaison des

paramètres QoS entre les niveaux de visibilité. L'opération de QoS mapping/translation s'exécute pendant la phase d'approvisionnement lors de l'établissement de la session, et également lorsqu'un changement se produit pendant la phase de l'exploitation lié à la mobilité ou à une dégradation de la qualité de service.

- QoS Negociator : offre une fonction de communication et de coordination verticale entre les niveaux de visibilité pour demander les ressources qui lui sont nécessaires ou les libérer. La communication horizontale assure le signalement d'activité ou de non activité du composant.

Nous avons présenté dans cette partie, d'une manière générale, le modèle organisationnel de l'architecture UBIS. Dans ce qui suit, nous présentons le composant de service autonomique (§V.2) qui est un élément clé du pilotage dynamique de la QoS de l'architecture UBIS car comme nous l'avons précisé précédemment tout est service.

V.2 Composant de service autonomique

V.2.1 Introduction

Dans l'objectif de satisfaire le contrat SLA établi entre les clients et les fournisseurs, et d'améliorer le «Service Delivery» au cours de la mobilité de la session, un pilotage dynamique de la QoS de bout en bout d'un service personnalisé est souhaité. Pour ce faire, nous présentons dans cette partie un composant de service autonomique, qui permet grâce à son agent de QoS de répondre aux besoins d'automatisation, d'autosuffisance et d'autogestion nécessaires pour garantir la continuité de service lors des différents types de mobilité. En effet, sa tâche principale est d'intégrer le contrôle et la gestion de la QoS au niveau de chaque composant de service de l'architecture et cela pour avoir un contrôle et une gestion décentralisé de la QoS. L'avantage de cette solution est de concevoir un composant de service capable de réagir dynamiquement et de manière autonome en temps réel suite à un changement dans le contrat de QoS durant la session de l'utilisateur, telle que la disponibilité. Ce composant jouera un rôle important dans l'obtention d'une composition de service plus flexible et avec une meilleure performance de QoS. Pour expliquer notre proposition, nous détaillons tout d'abord dans la section (§V.2.2) l'aspect architectural et fonctionnel du

composant de service autonomique. Ensuite, nous expliquons dans la section (§V.2.3) le modèle QoS qui est la base de l'autogestion de composant de service. Et finalement, nous expliquons dans la section (§V.2.4) les mécanismes appliqués au cours de chaque étape opérationnelle, qui permettent à un composant de service d'autocontrôler et d'autogérer ses propres ressources, pour avoir une réaction plus flexible durant la session de l'utilisateur, et cela pour maintenir le «service Delivery» avec la QoS requise.

V.2.2 Structure d'un composant de service autonomique

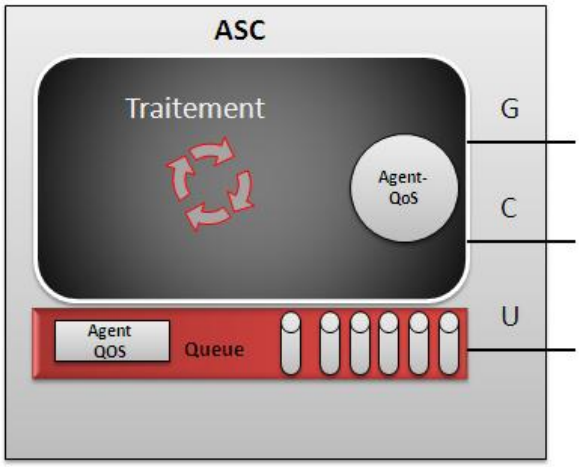

Figure 17: Composant de service autonomique

L'architecture orientée services (SOA) joue un rôle principal dans la création rapide des applications grâce à une composition des composants de service indépendants. Ces composants fournis par différents fournisseurs peuvent offrir des fonctionnalités identiques mais avec des capacités de QoS différentes. Le composant de service autonomique «ASC» que nous proposons est une nouvelle vision de l'architecture SOA. Grâce à son agent de QoS, il a les capacités de contrôle et de gestion de la QoS lors de la session de l'utilisateur pour améliorer le service delivery.

Le composant de service autonomique a les caractéristiques suivantes: Stateless, Mutualisable, Autonome et Auto-gérable.

- *Stateless*: un composant de service est stateless s'il effectue les mêmes traitements (opérations) pour toutes les demandes provenant de différents utilisateurs et cela sans conserver les données ou l'état spécifique à chacun d'eux. Cette caractéristique est importante dans le cas où il est nécessaire de faire un remplacement dynamique d'un composant de service qui a subi une dégradation de la QoS au cours de la session d'un utilisateur, pour que les données et l'état relatifs à une requête précédente ne perturbent pas le traitement de la nouvelle requête.
- *Mutualisable*: le composant de service est mutualisable parce qu'il est conçu pour traiter plusieurs demandes provenant de différents utilisateurs en même temps selon ses capacités. Pour assurer le contrôle et la gestion de ce partage de ressources, on associe une file d'attente (Figure 17) au niveau du plan d'usage du composant de service, cette file d'attente permet de provisionner toutes les requêtes acceptées des utilisateurs. Les ressources relatives à la file d'attente sont contrôlées et gérées par un agent de QoS intégrés dans la file.

Le composant de service est autonomique parce qu'il est à la fois autonome et auto-gérable.

- *Autonome*: Le composant est autonome parce qu'il est fonctionnellement indépendant c'est-à-dire il est auto-suffisant et il n'a pas besoin d'autres composants de service pour accomplir ses fonctionnalités. L'intérêt de cette indépendance fonctionnelle entre les composants de service est de faciliter le changement d'un composant de service par un composant de service ubiquitaire en cas de détérioration de la QoS ou de dysfonctionnement pendant l'exploitation et cela sans impacter le service global demandé par l'utilisateur.
- *Auto-gérable*: Le composant de service est auto-gérable parce qu'il surveille sa propre QoS et il gère aussi ses états liés à l'utilisation des ressources pendant la session utilisateur. Le composant de service a, à un instant t, un de ces quatre états Indisponible, Disponible, Activable, Activé (Figure 18).
  - L'état indisponible signifie que le composant de service est temporairement ou définitivement inaccessible,
  - L'état disponible signifie que le composant de service peut être accessible,
  - L'état activable signifie que le composant de service est prêt à être activé,

- Et l'état activé signifie que les ressources du composant de service sont en cours d'utilisation. Lorsque le composant de service est activé, son agent de QoS, inclus dans son plan de gestion, mesure et contrôle le contrat de QoS établi entre l'utilisateur et le fournisseur de service.

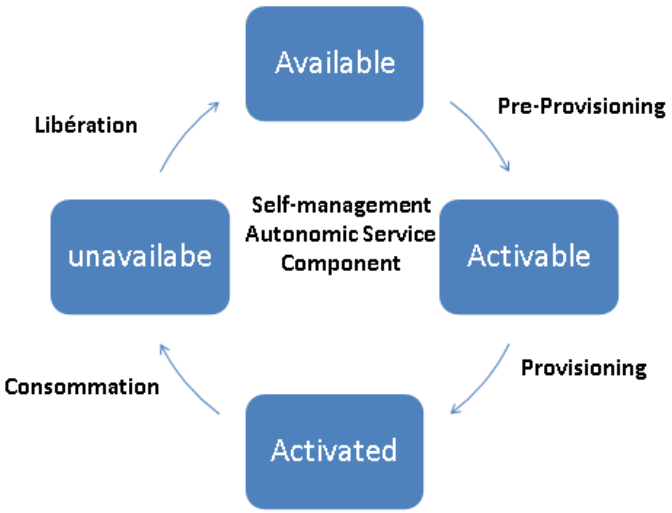

Figure 18: les états d'un composant de service autonomiques

Le composant de service autonomique (ASC) est structuré selon 3 plans: contrôle, usage et gestion:

- Le plan d'usage contient les fonctions principales réalisées par le composant de service pour rendre le service, il comprend tous les mécanismes utilisés pour le traitement d'une demande pendant la phase de consommation.
- Le plan de contrôle contient tous les mécanismes, utilisés d'une manière asynchrone durant la session de l'utilisateur, qui permettent de réserver les ressources du composant de service nécessaires au traitement de la demande de l'utilisateur.
- Le plan de gestion contient les mécanismes appliqués d'une manière asynchrone sur les données, pour gérer et contrôler les ressources du composant de service pendant la phase de consommation.

Un composant de service dispose de trois interfaces : usage, contrôle et gestion. Pour obtenir des processus transverses aux trois plans afin d'avoir l'automatisme nécessaire à la

dynamicité et à la continuité requises par la session, nous avons une interface convergée entre les différents plans du composant de service. Chaque interface convergée définit un jeu d'opérations et d'actions entre deux interfaces élémentaires pour avoir plus de dynamicité, de flexibilité et d'adaptabilité auprès du service à rendre.

- La flexibilité est obtenue grâce à la convergence des deux plans: usage et contrôle, pour prendre en considération les préférences temporelles et spatiales de l'utilisateur au cours de la session. Cette convergence permet d'assurer une surveillance des paramètres de performance pour garantir la QoS pendant la phase d'exploitation avec une réaction en temps réel dans le cas d'une mobilité ou bien d'un changement lié aux préférences de l'utilisateur.

- L'adaptabilité est obtenue grâce à la convergence du plan d'usage et du plan de gestion. Son enjeu majeur est d'assurer l'interopérabilité entre les composants de service dans le cas où un changement se produit au cours de la session. Elle a pour objectif la réservation des ressources en adéquation avec la QoS requise pour le traitement de la demande des utilisateurs. Cette convergence offre une coordination entre les informations du plan d'usage et celles du plan de contrôle pour appliquer les adaptations nécessaires à la continuité de service avec la qualité de service requise.

- Et la dynamicité est obtenue grâce à la convergence du plan de gestion et du plan de contrôle. Elle permet de fusionner les informations de gestion (détection de défaillance, localisation de dysfonctionnement, dégradation QoS) et l'automatisme du plan de contrôle (signalisation) pour réagir le plus rapidement possible pendant la phase de l'exploitation et maintenir la conformité du SLA. Grâce à cette convergence l'agent de QoS intégré dans le plan de gestion joue également un rôle dans le contrôle de la QoS au niveau du plan de contrôle.

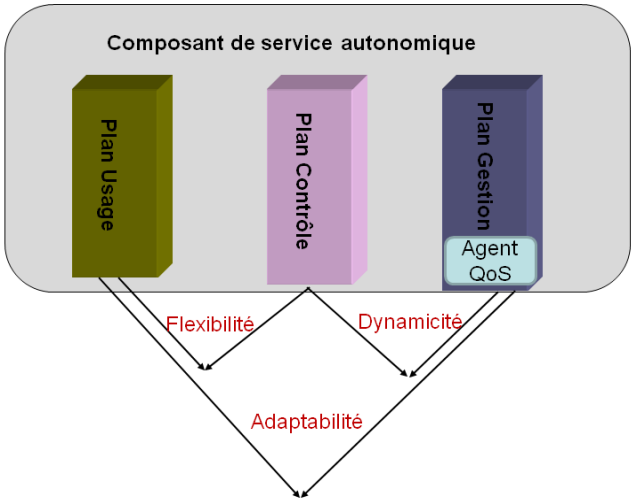

Figure 19: Convergence des plans

V.2.3 Le Modèle de QoS

Pour évaluer le comportement de QoS de bout en bout au cours de la session utilisateur, il est nécessaire d'avoir une expression homogène de la qualité de service d'un composant de service. Pour ce faire, chaque composant de service instancie un modèle de QoS pour gérer les ressources service en temps réel et leurs possibles dégradations. Cette solution semble efficace pour répondre au besoin de l'automatisation et de distribution du contrôle et de gestion de la QoS au niveau service de l'architecture.

Pour cette raison nous avons proposé un modèle de QoS qui décrit la QoS d'un composant de service d'une manière homogène et uniforme. Le comportement de chaque composant de service est reflété par les paramètres de QoS mesurables qui sont classés selon un vecteur de quatre critères: Disponibilité, fiabilité, délai et capacité.

- *Disponibilité* «**A**» indique le taux d'accessibilité d'un composant de service, elle indique le pourcentage de temps qu'un composant est en mesure d'accepter des demandes

$$A = 1 - U / T$$

Avec **U** et **T** qui représentent respectivement le nombre de demandes qui ont été rejetées par un composant de service, et le nombre total de demandes envoyées sur une période de temps.

- *Fiabilité* «**F**» représente la capacité d'un composant de service à fonctionner correctement sans modification de l'information traitée, elle indique le pourcentage des invocations réussies sur une période de mesure.

$$F = 1- R/ T$$

Avec **R** qui indique le nombre de demandes qui ont échoué pendant une période de temps.

- *Délai* **D** représente la durée moyenne pour traiter une demande par un composant de service.
- *Capacité* **C** représente le taux de charge maximal supporté par un composant de service.

Ces critères de QoS sont tous nécessaires et suffisantes pour qu'un composant de service s'autocontrôle et s'autogère. Ces critères sont évalués à travers trois types de valeurs mesurables : les valeurs de conception, les valeurs courantes et les valeurs seuils.

- Les *valeurs de conceptions* sont déterminées au moment de la conception du composant de service, elles définissent les capacités maximales de traitement d'un composant de service.
- Les *valeurs courantes* sont utilisées pendant la phase de l'exploitation pour surveiller le comportement d'un composant de service au cours du traitement des demandes.
- Les *valeurs seuils* indiquent les valeurs limites à ne pas dépasser par un composant de service pour qu'il assure son traitement normalement.

Ces informations de QoS sont utilisées pour prendre les décisions pendant la phase de l'exploitation. En effet, pour prendre les bonnes décisions au cours de la session de l'utilisateur, au bon endroit et au bon moment, il est nécessaire d'avoir une représentation efficace du monde réel.

Pour cette raison, nous avons besoin d'avoir une structure d'informations uniforme qui contient à la fois la description du composant de service (fonctionnel) et aussi la connaissance de son comportement (non-fonctionnel QoS). Cette description est fournie par le modèle

informationnel qui contient les différents profils à solliciter durant les différentes phases opérationnelles d'un composant de service: provisioning et exploitation

- Pour la phase du pré-provisioning, nous avons le profil des ressources, qui contient les valeurs QoS de conceptions définies à la phase conceptuelle du composant de service.
- Durant la phase d'exploitation, le profil Real Time est instancié en temps réel pour avoir une gestion dynamique de la QoS, il contient les valeurs de QoS courantes qui seront comparées aux valeurs de QoS seuils pour contrôler le comportement d'un composant de service pendant l'usage.

V.2.4 Les mécanismes de gestion de la QoS

Dans l'objectif d'améliorer le «service Delivery», nous avons intégré des mécanismes de gestion de la QoS dans le composant de service autonomique. Nous avons proposé des mécanismes pour chaque phase opérationnelle:

Durant la phase du pré-provisioning, il faut sélectionner un composant de service en se basant sur les valeurs de conceptions. Pour cela, au moment de l'initiation de la session, chaque ASC procède par un contrôle d'admission de QoS « QAC» pour vérifier s'il dispose des capacités requises pour traiter une nouvelle demande d'un nouveau utilisateur. Cette étape permet d'identifier et de sélectionner les composants de services qui répondent aux besoins des utilisateurs en terme de QoS, parce qu'il existe plusieurs composants de service ubiquitaires qui offrent les mêmes fonctionnalités mais avec des QoS différentes.

Durant la phase du provisioning, le composant de service réserve les ressources nécessaires pour traiter la demande de l'utilisateur. Comme nous l'avons souligné précédemment, chaque composant de service contient une file d'attente qui stocke les requêtes acceptées de différents utilisateurs parce qu'il est mutualisable. Pour contrôler ce partage de ressources, nous devons appliquer un contrôle d'admission de QoS «QAC» pour accepter une nouvelle demande dans la file d'attente d'un composant de service. Le rôle du «QAC» est de déterminer si une nouvelle requête peut être acceptée dans la file d'attente, sans violer les SLA des requêtes qui sont déjà acceptées.

Durant la phase de consommation, il faut contrôler et gérer la QoS d'une façon dynamique et continue. Notre vision d'autogestion de la QoS est basée sur l'agent de QoS intégré dans le plan de gestion du composant de service, et qui mesure continuellement les ressources QoS relatives à son traitement interne. Chaque composant appartient à une communauté de service virtuel «VSC» qui contient des composants de service qui ont les mêmes fonctionnalités et QOS. En cas de dégradation de la QoS, l'agent QoS envoie un évènement à sa communauté VSC pour qu'elle assure son remplacement par un composant équivalent, et par conséquent adapter la session de l'utilisateur sans causer une détérioration du service fourni.

Chaque file d'attente d'un composant de service appartient à une communauté virtuelle des files d'attente «VQC» qui regroupe des files d'attente qui ont les mêmes ressources, si l'agent QoS de la file d'attente détecte qu'une requête a pris plus temps que prévu pour son traitement (délai), il notifie sa communauté VQC pour réapprovisionner cette requête dans une autre file d'attente pour ne pas impacter le temps de traitement des autres requêtes qui sont déjà provisionnées dans la file.

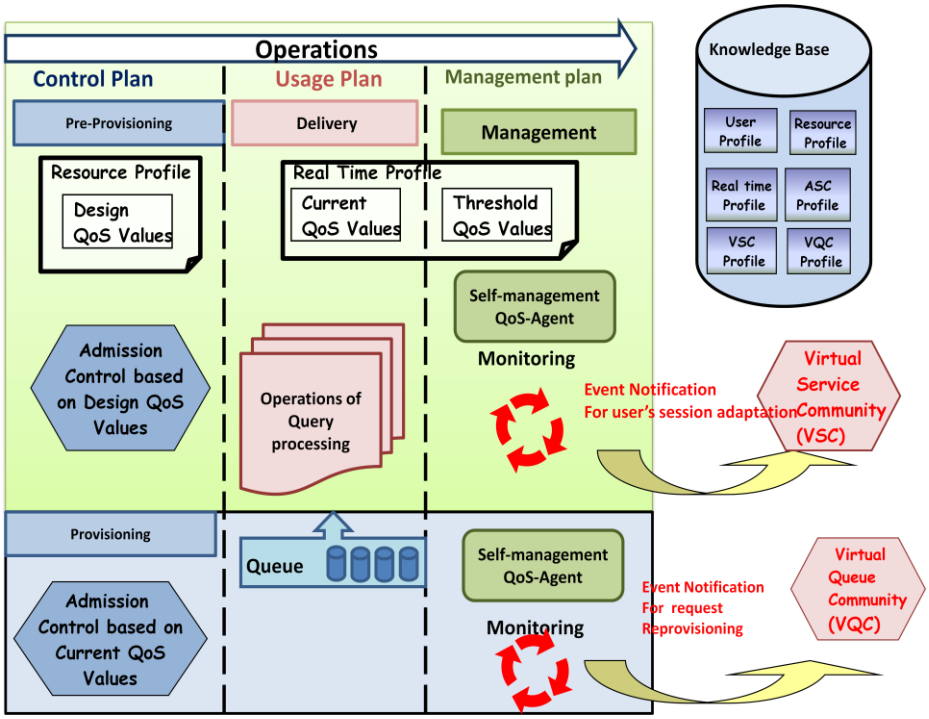

Figure 20: Mécanismes de gestion de la QoS

Nous détaillons dans ce qui suit via un automate, les mécanismes de gestion de la QoS utilisés dans chaque plan du composant de service autonomique au cours des différentes phases opérationnelles: le pré-provisioning, le provisioning, et le delivery.

Pendant la phase du pré-provisioning (Figure 21), chaque ASC procède par un contrôle d'admission pour vérifier s'il a les capacités requises pour traiter une nouvelle demande d'un utilisateur. Tout d'abord, il reçoit dans son plan de contrôle un message de signalisation pour attacher une nouvelle session de service. Puis, il incrémente la variable «Attach» à «Attach + 1» pour garder la traçabilité du nombre de sessions de service qui sont attachées à ce même service. Ensuite, l'agent QoS vérifie la QoS statistique qui est basée sur les valeurs de QoS de conception, afin de contrôler son attachement à une nouvelle session de services, car un composant de service peut être partagé entre plusieurs utilisateurs parce qu'il est mutualisable. Si le résultat est positif "OK", l'ASC change son état à activable dans le cas où il n'est pas précédemment rattachées à une session de service d'un autre utilisateur. Ensuite, il ajoute l'identifiant de la session utilisateur «ID VPSN» dans son profil dans la base de connaissance L'INFOWARE. Si le résultat est négatif "NOK", il faut décrémenter la variable «Attach» à «Attach-1" et ensuite le système se charge de trouver un autre composant de service ayant les mêmes fonctionnalités et une QoS équivalente pour répondre à la demande de l'utilisateur.

Pendant la phase de Provisioning (Figure 21), l'ASC reçoit la transaction dans son plan de contrôle, puis il incrémente la variable J à «J+1». Ensuite, il vérifie la QoS dynamique en se basant sur la QoS courante de la file d'attente afin de contrôler l'acceptation de la transaction de l'utilisateur dans la file d'attente. Si le résultat est positif, il provisionne la requête dans la file d'attente et il change son état de activable à activé dans le cas où il n'y a pas de requêtes provisionnées auparavant dans la file d'attente (c'est-à-dire J = 1), sinon il décrémente la variable de J à J-1.

Pendant la phase de consommation un composant de service peut ne pas continuer à fonctionner normalement ou à répondre aux exigences de l'utilisateur en termes de QoS. C'est pour cette raison que nous proposons des mécanismes pour surveiller et contrôler la QoS au niveau de chaque composant de service. Notre concept repose sur un agent de QoS intégré dans le plan de gestion de chaque composant de service. La (**Erreur ! Source du renvoi**

introuvable. Figure 21) montre les différents mécanismes utilisés lors de la phase de la consommation pour maintenir le service pour l'utilisateur avec le niveau de QoS requis.

Si l'ASC prend plus de temps pour traiter une requête (Timer 4 a expiré), cela peut donc affecter le temps de traitement des autres requêtes qui sont déjà provisionnées dans sa file d'attente. C'est pour cette raison qu'on utilise un agent de QoS pour surveiller et contrôler la QoS de la file d'attente. Lorsque l'agent QoS détecte une violation de contrat de QoS, il doit aviser la VQC (Virtual Queue Community) pour renvoyer la requête dans une autre file d'attente ubiquitaire. Il convient de mentionner que chaque composant de service appartient à un VSC (Virtual Service Community), qui contient un ensemble de composants de service autonomique ayant les mêmes fonctionnalités et une QoS équivalente, et aussi chaque ASC dispose d'une file d'attente faisant partie d'une communauté virtuelle de file d'attente VQC (Virtual Queue Community) qui contient des files d'attente ubiquitaires.

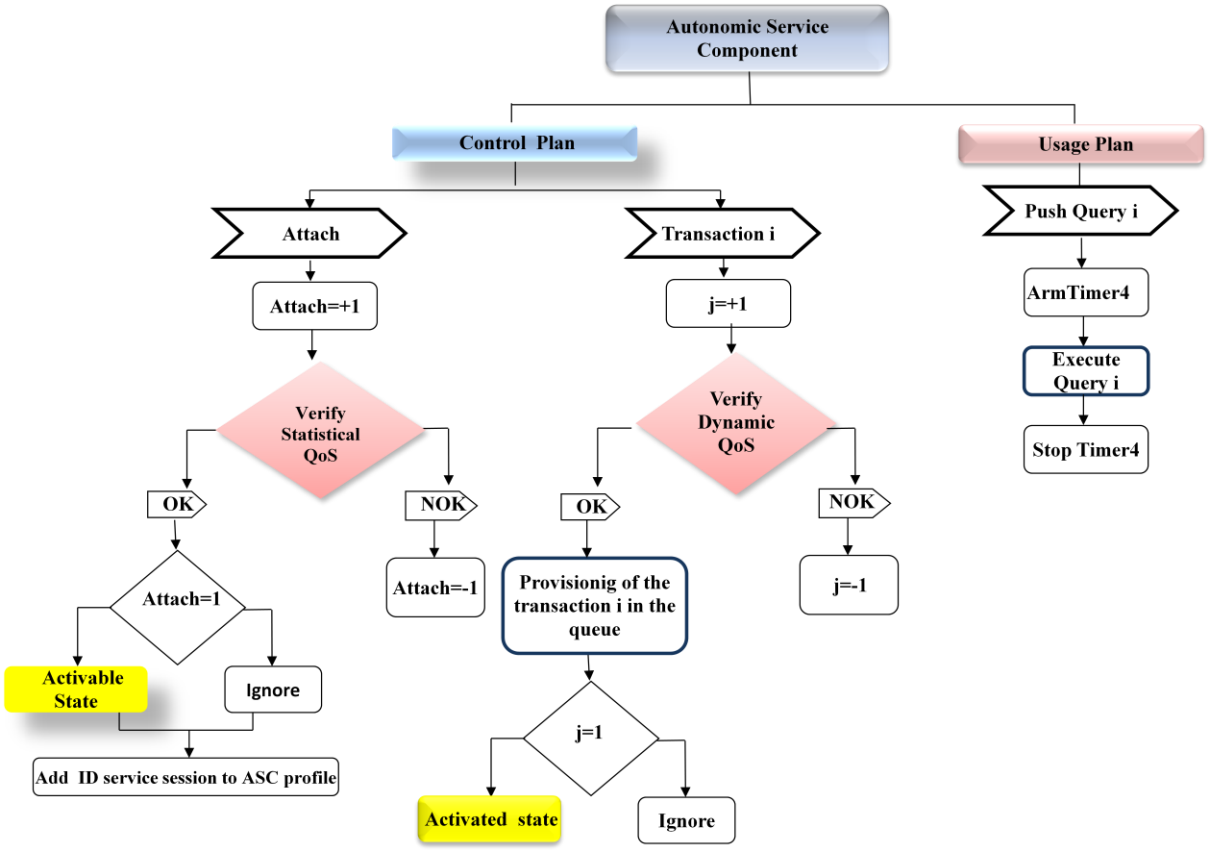

Figure 21: Automate du composant de service (plan d'usage et de contrôle)

Durant la consommation (Figure 22), l'agent QoS de chaque ASC qui compose le service global de l'utilisateur compare la QoS courante avec les valeurs de QoS seuils. Si le résultat est positif, l'agent QoS envoie un événement «IN Contract» à la communauté «VSC» pour notifier qu'il respecte toujours son contrat de QoS. Sinon, dans le cas où l'agent QoS détecte une dégradation de la QoS il envoie un événement «Out contrat» (Armer Timer2 et Timer3) à la communauté VSC afin de lancer l'algorithme de remplacement du composant de service qui est en cours d'utilisation par un composant de service ubiquitaire (Arrêter Timer2). Lorsque le Timer 3 expire, l'ASC change son statut de disponible à indisponible et il quitte sa communauté. Puis il rejoint une nouvelle communauté VSC selon sa QoS courante. Il change par conséquent son état de indisponible à disponible. Toutes ces opérations sont transparentes pour l'utilisateur final et sont effectuées automatiquement par le système lors de la session utilisateur.

Chaque ASC participe à la gestion de sa communauté VSC; il notifie régulièrement les autres membres sur son état de QoS (In contrat / Out contrat). Pour ce faire, si l'ASC reçoit un Out Contract il met son FlagOUT= 1 et il informe sa communauté VSC pour exclure l'ASC qui est hors contrat de la communauté et de procéder par la suite à son remplacement afin de maintenir la communauté VSC. Si l'ASC reçoit un «IN contrat», il arrête leTimer1 qui a été armé en attendant de recevoir un «IN contrat». Dans le cas où le timer 1 expire sans recevoir un IN contrat, l'ASC doit tout d'abord vérifier son FlagOut. Si le FlagOut= 1, cela signifie que l'ASC a déjà reçu un Out Contract. Sinon, il avise la VSC sur une communication défectueuse de l'ASC.

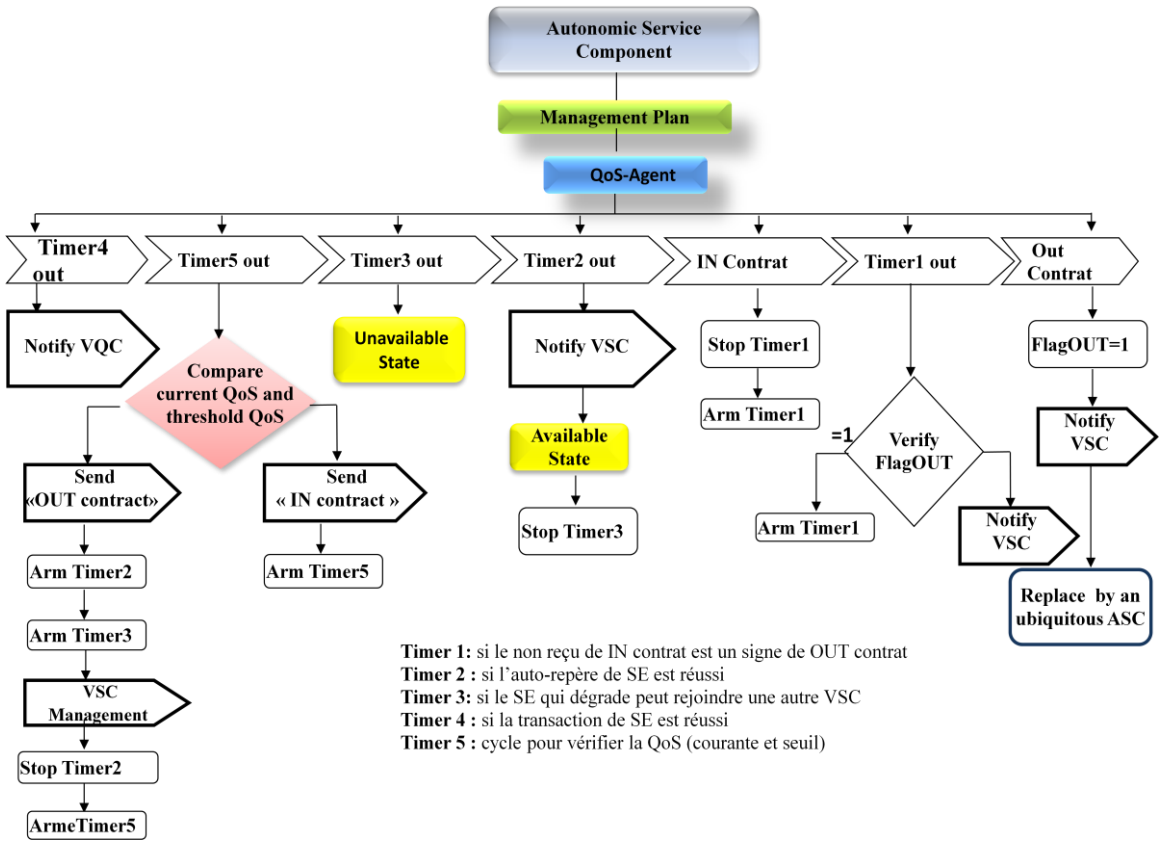

Figure 22:Automate du composant de service (plan de gestion)

V.3 Dimension protocolaire

V.3.1 Introduction

L'établissement, la modification et la maintenance d'une session unique et sans couture dans un contexte hétérogène et quel que soit le type de mobilité (utilisateur, terminal, réseau et service) est un point important pour répondre au besoin de l'utilisateur. Les différents types de mobilité qui peuvent se produire au cours de la session «user-centric», nécessitent l'exécution d'interactions plus complexes entre les éléments de service appartenant aux différents niveaux de visibilité en vue de rendre le service attendu avec la qualité de service requise.

La composition des services s'appuie sur l'agrégation modulaire de plusieurs services élémentaires issus d'environnement et de plate-forme hétérogènes. Cette composition doit être hautement dynamique, c.-à-d. activable à la demande des utilisateurs et elle doit d'une part, prendre en compte les besoins et contraintes multiples et évolutifs dans le temps en fonction du contexte ciblé, et d'autre part traiter dynamiquement les évolutions de ce contexte liées à la mobilité ou à une dégradation de la qualité de service.

Pour ce faire, nous structurons l'échange de données selon la dimension protocolaire qui se définit par l'intersection de trois éléments principaux les interfaces, les protocoles de communication et le plan d'appartenance. Elle inclut tous les mécanismes d'échanges, d'interactions et de coopérations entre les éléments pour assurer d'une part la composition de service de manière efficace et automatisée et d'autre part pour satisfaire le besoin d'une «session unique» sans couture et sans coupure. Le maintien de ce type de session impose de fortes interactions entre les acteurs afin de prendre en compte dynamiquement les changements, tant du côté de l'usager liés à la mobilité ou au changement de ses préférences que du côté de l'environnement d'exécution (QoS du réseau et des composants de services).

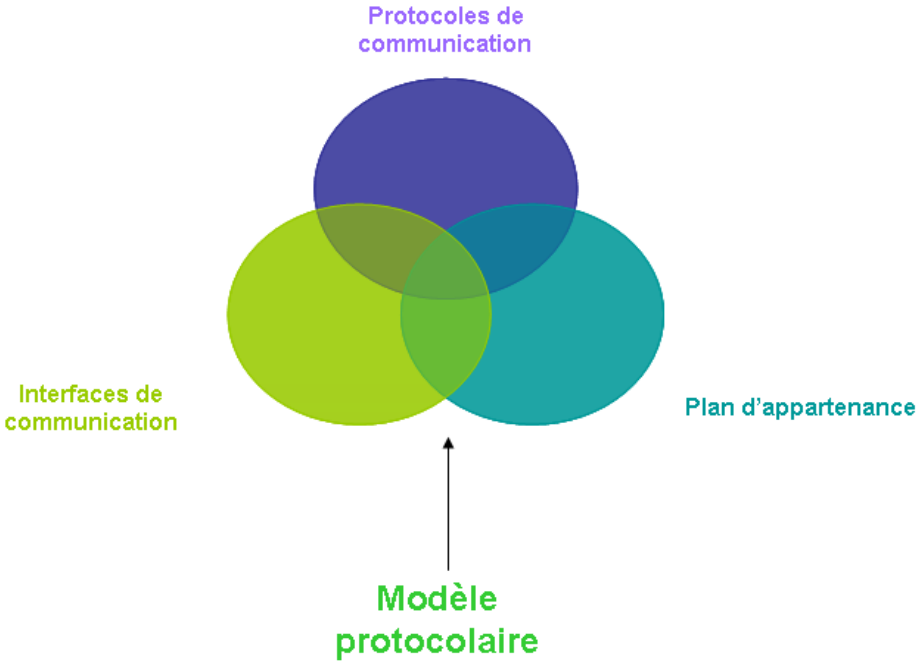

Figure 23: Modèle protocolaire

- ***Les protocoles de communication***

Chaque entité dans l'architecture UBIS rend un service qui peut être de type usager, contrôle ou gestion. Pour assurer la communication entre ces différentes entités, il existe des protocoles spécifiques pour chaque type de service qui permettent de structurer l'échange des informations entre les entités communicantes à travers les interfaces associées (gestion, contrôle et usage).

Les protocoles décrivent le comportement de communication externe des services qui peuvent être établies entre deux éléments de services. Pour assurer une interopérabilité correcte et automatisée, il existe des protocoles relatifs pour acheminer les données entre les composants de service faiblement couplés à travers les interfaces correspondantes de chaque plan (plan d'usager, plan de contrôle et plan de gestion). En outre, la composition de service permet de créer des services personnalisés à l'aide des composants de service issus des environnements et plates-formes hétérogènes. Il faut donc définir les protocoles d'interactions pour accomplir la composition de service et maintenir la session pendant les différents changements qui

peuvent se produire suite à la mobilité ou à une dégradation de la qualité de service d'une manière simplifiée et automatisée.

Les protocoles de communication permettent de structurer les messages échangés, ils représentent l'ensemble des règles permettant de définir un type de communication particulier. Ils adressent les aspects liés au transport des messages de type usage, contrôle ou gestion. Pour cela, il faut spécifier la nature des messages échangés et le mode.

Généralement, ces messages sont des requêtes (confirmées ou non confirmées) ou des notifications. Grâce à ce type de message, les entités permettent de réaliser une tâche en traitant les messages reçus ou en prenant l'initiative lorsqu'elles ne sont pas capables de fournir le service demandé. Pour le plan d'usage, on distingue deux autres natures de message:

- Les Interactions (doublet <question/réponse>) permettent aux entités de l'architecture UBIS de dialoguer entre elles au cours de la session.
- Les transactions (triplet <question, traitement, réponse> sont à la demande, elles répondent aux propriétés «ACID». Une transaction peut correspondre à une procédure SQL utilisée par les bases de données ou à une requête HTTP.

Deux modes de communication peuvent être envisagés:

- Mode connecté : L'entité émettrice et l'entité réceptrice du message doivent être actives en même temps pour vérifier si les données envoyées sont effectivement reçues.
- Mode non connecté: L'entité émettrice et l'entité réceptrice du message ne sont pas actives en même temps. Ce qui n'implique pas d'être connecté en permanence, ni de recevoir immédiatement une réponse. Ce mode correspond à des architectures faiblement couplée, c'est-à-dire que chaque composant de service n'impose pas à l'autre une interface de communication, c'est au message lui-même d'encapsuler les données et le contexte devenant ainsi une entité autonome.

- **Le plan d'appartenance**

L'architecture UBIS est organisée selon une structuration en 3 plans dans laquelle les piles protocolaires sont autonomes. Le plan usager pour les données, le plan de contrôle pour la signalisation et le plan gestion pour la supervision interne.

Nous proposons pour cette dimension protocolaire une articulation autonome des 3 plans grâce à la mise à disposition des informations:

- le plan de gestion pour les informations de surveillance durant l'exploitation,
- le plan de contrôle pour celles du provisioning et
- le plan d'usage pour la consommation des ressources transaction par transaction.

*Plan de gestion***:** C'est dans le plan de gestion que sont décidées, évaluées et modifiées selon le besoin les politiques appliquées par le plan de contrôle. Ce plan regroupe les opérations et fonctions y compris mécanismes et protocoles effectués sur les flux de données de manière asynchrone. Il regroupe tous les flux de gestion entre les entités communicantes à des fins de qualité de service de bout en bout.

Les flux de gestion qui ont pour rôle:

- L'authentification, autorisation, et la configuration du service demandé.
- La notification du statut de la QoS (In/Out contrat)
- Le transfert des règles de politiques QoS.

Plan de contrôle: Le plan de contrôle contient les opérations exécutées de manière synchrone sur les données applicatives émises ou reçues. C'est dans le plan de contrôle qu'est réalisée la composition des mécanismes permettant d'appliquer la politique choisie pour offrir le service attendu. Le plan de contrôle regroupe également l'ensemble des flux de signalisation nécessaires à la réservation de ressources et à la transmission des données relatives à l'établissement, la modification ou la terminaison d'une session entre deux entités.

Les flux de contrôle ont pour rôle:

- Le routage basé sur les informations QoS.
- Le contrôle d'admission basé sur l'état des ressources disponibles et la négociation de service de QoS.

– La signalisation de QoS pour la réservation des ressources et la négociation de service QoS.

*Plan Usage***:** Le plan usage a en charge le transport des informations utilisateurs, il regroupe tous les mécanismes relatifs au transfert des données du service dans le réseau. Ce plan comprend également les traitements de contrôle d'erreurs et de contrôle de flux qui sont appliqués parallèlement au transfert des données.

- **Les interfaces**

L'échange de message s'appuie sur des protocoles de communications selon le type d'informations échangées. Pour chaque plan (plan usager, plan de contrôle et plan de gestion), il y a des protocoles correspondants pour transmettre les informations à travers les interfaces associées.

Notre but est d'utiliser les protocoles existants pour transporter les informations des interfaces convergées. En effet, pour la fusion des actions, à l'interface, il faut identifier les différents protocoles possibles à chaque interface convergée et proposer des protocoles améliorés ou modifiés afin de simplifier les messages des différents protocoles et réduire le temps de traitement.

Notre objectif est donc d'assurer la QoS de bout en bout pour la continuité de session dans le NGN en intégrant les trois plans pour plus de dynamicité, de flexibilité et d'adaptabilité, mais non pas en rajoutant des protocoles dans les interfaces intégrées, mais en faisant converger ceux qui existent à travers l'information comme nous l'avons expliqué ci-dessus.

- **Adaptabilité (convergence plan usage et plan de contrôle)**

Quelle politique mettre en place au niveau des interfaces pour avoir plus d'adaptabilité? Quel est l'intérêt de la convergence du plan usage et du plan de contrôle?

Les capacités d'adaptabilité sont obtenues grâce à la convergence des deux plans usage et contrôle. L'adaptabilité est un enjeu majeur pour assurer l'interopérabilité des sous réseaux hétérogènes et la réservation des ressources en adéquation avec la QoS de chacun des sous réseaux. L'adaptabilité est nécessaire pour prendre en compte les changements induits par les

différentes mobilités qui conduisent à un changement de réseaux d'accès ou réseaux cœur avec des QoS différentes. Donc, elle permet selon les besoins d'affiner la qualité de service. Pour cela, il faut coordonner entre les informations du plan d'usager et celles du plan de contrôle pour appliquer les adaptations nécessaires pour la continuité de connectivité avec la qualité de service requises.

Par exemple, dans l'objectif d'optimiser une qualité de service de bout en bout il faut effectuer une adaptation pour assurer l'interopérabilité de la classification du trafic et du marquage de paquets dans les réseaux hétérogènes dans le NGN.

Dans cette interface convergée, il se trouve les requêtes du plan usage pour le contrôle d'admission et les messages de QoS signalisation. Les protocoles possibles pour cette interface seraient des «protocoles de signalisation» comme NSIS ou SIP, etc.

- **Flexibilité (convergence plan usager et plan de gestion)**

Quelle est la politique à mettre en place au niveau des interfaces pour avoir plus de flexibilité au cours de l'exploitation?

Les capacités de flexibilité sont obtenues suite à la convergence des deux plans gestion et usage pour tenir en considération les préférences spatiales et temporelles de l'utilisateur. Cette convergence permet d'assurer une surveillance des paramètres de performance de service pendant l'exploitation avec une réaction en temps réel suite à un changement lié à la mobilité ou bien aux préférences de l'utilisateur pour maintenir le service.

Dans cette interface convergée se trouve les informations relatives à l'utilisateur et les informations de gestion sur la tarification ce qui permet plus de flexibilité, et l'utilisateur a la meilleure connexion avec la tarification la plus appropriée.

- **Dynamicité (Convergence plan de gestion et plan de contrôle)**

Quelle est la politique à mettre en place au niveau des interfaces pour faire face au dynamisme?

La convergence du plan de gestion et du plan de contrôle répond aux capacités de dynamicité. Elle permet de fusionner les informations de gestion (détection de défaillance, localisation de

dysfonctionnement, dégradation QoS) et l'automatisme du plan de contrôle (signalisation) pour réagir le plus rapidement possible pendant la phase de l'exploitation et maintenir la conformité de SLA.

Grâce à la convergence CG (contrôle-Gestion), on transfert les informations de contrôle liées à la configuration des services et les messages de surveillance du plan de gestion au plan de contrôle. Et puis, le plan de contrôle fait le routage ou re-routage correspondant aux informations de QoS et il renvoie le feed-back au plan de gestion. Les protocoles possibles utilisables pour cette interface sont les protocoles de gestion Diameter ainsi que les protocoles de contrôle SIP/SDP etc.

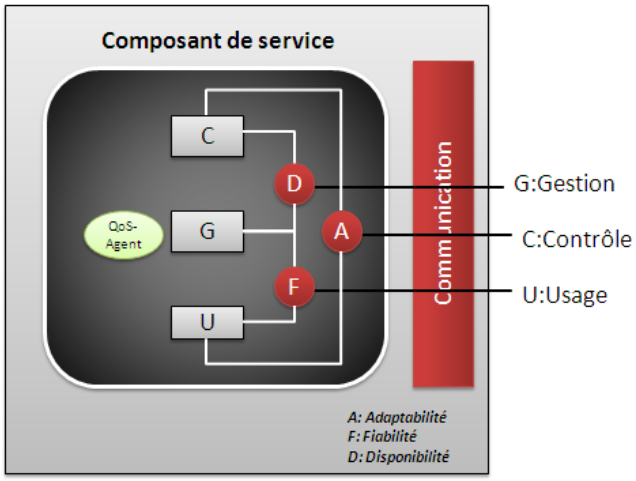

Figure 24: Composant de service (Interfaces convergées)

V.3.2 La session UBIS

(a) Les besoins de la Session UBIS

Actuellement, IMS (IP Multimedia Subsystem) spécifié par la 3GPP (Third Generation Partnership Project) propose une négociation de QoS pour déterminer la meilleure configuration des services et également pour assurer la réservation des ressources réseau, ce qui garantit les performances du réseau pour chaque service multimédia. IMS fournit les mécanismes de gestion de QoS qui visent à offrir aux utilisateurs finaux une meilleure prestation des médias à travers plusieurs terminaux et réseaux d'accès (mobilité d'utilisateur et mobilité du terminal). Les procédures d'IMS pour la négociation de la QoS sont basées sur le

protocole de signalisation SIP/SDP spécifié par l'IETF (Internet Engineering Task Force). En effet, IMS initie le processus de la création des sessions media via les invitations du protocole de signalisation SIP qui transporte le protocole SDP (Session Description Protocol) pour permettre une négociation des descriptions media entre les participants afin d'assurer le Media Delivery au niveau du réseau. Les solutions actuelles du «Media Delivery» offrent un ensemble de mécanismes et protocoles pour permettre uniquement la négociation de la QoS et la réservation des ressources réseau. Donc, le «Service Delivery» ne tient pas compte de la négociation et de la gestion des ressources service au niveau service de l'architecture, alors que des problèmes surgissent durant la session à cause du comportement des services au niveau de la plate-forme de services. Ces solutions actuelles ne sont plus suffisantes car les utilisateurs d'aujourd'hui veulent accéder à leurs services personnalisés dans un contexte hétérogène et mobile. Les fournisseurs devraient donc garantir à l'utilisateur une session unique et continue pour offrir une continuité de service avec une QoS de bout en bout. En fait, le bout en bout suppose que, hormis les ressources réseaux, les ressources services devraient être prises en compte en tant que ressources importantes lors de la session de l'utilisateur. Au niveau de la couche réseau de nombreux protocoles tel que SIP/SDP sont utilisés pour négocier la QoS des ressources réseaux. Créer et maintenir ce type de session nécessite de fortes interactions plus flexibles qui agissent sur la couche de service, par analogie à ceux de la couche réseau. C'est pourquoi nous proposons une extension de la portée du protocole SIP existant à la couche service, nommé SIP +, qui sera utilisé pour négocier la QoS des services personnalisés à la phase d'initialisation de la session de service et aussi de renégocier la QoS lors de la phase d'exploitation pour remédier au problème de dégradation de la QoS causé par la mobilité.

Ce protocole permet de créer, modifier et terminer la session de service d'un utilisateur basée sur la composition des services dans une architecture trans-organisationnelle. Dans cette partie sont exposés les atouts intéressants des spécifications protocolaires attachées au protocole SIP+ qui sont utiles à la composition de services. Le protocole SIP + permet d'assurer le contrôle de la session pendant la phase d'établissement de la session (création du VPSN) et la gestion de la session par le biais du VSC (Virtual Service Community). Pour ce faire, Il faut enrichir les informations véhiculées par les méthodes SIP en rajoutant les informations nécessaires pour le contrôle et la gestion de la QoS au niveau service. Ces

informations permettent de négocier la QoS pendant la phase d'initialisation de la session de service et aussi de renégocier la QoS lors de l'exploitation. Il existe plusieurs méthodes SIP, nous utilisons uniquement la méthode INVITE pour la négociation de SLA, la méthode NOTIFY pour la renégociation de QoS et BYE pour la libération des ressources.

L'établissement d'une session UBIS se déroule en trois étapes : l'ouverture de la session de services pour une personnalisation des services, l'initialisation de la session de services et la création du VPSN.

(b) Les étapes d'une session UBIS

(b.1) Ouverture de la session de services: Personnalisation des services

Lorsque l'utilisateur est autorisé à ouvrir sa session UBIS, il compose à travers une IHM « Interface Homme Machine» des services exposables hétérogènes en fonction de ses besoins (fonctionnel et non-fonctionnel (QoS)) qu'il va utiliser durant sa session. La sélection des services exposables se fait soit à partir du catalogue des services {SExp α, SExp β} stockés dans le profil utilisateur ou bien par la découverte des services existants dans l'environnement de l'utilisateur {SExp γ} (**Erreur ! Source du renvoi introuvable.**Figure 23).

Durant la phase de la découverte des services, on prend en considération les informations de géolocalisation de l'utilisateur (longitude: LO, latitude: LA) pour favoriser les services localisés dans le réseau ambiant de l'utilisateur afin de lui offrir des services avec une valeur ajoutée de son environnement. Les informations de géolocalisation sont obtenues en utilisant un terminal localisé par le biais du GPS, WIFI, GSM, IP ou bien d'autres techniques. Les informations de géolocalisation obtenues sont stockées dans le terminal ou bien dans le serveur de services.

Ensuite, le terminal génère le Workflow avec la composition des services exposables {SExp α @ provider1, SExp β @ provider2, SExp γ @ provider3}, la logique de service et également la localisation de l'utilisateur pour initier la session UBIS de l'utilisateur.

Ces informations sont envoyées via le protocole de signalisation SIP+ en passant par IMS à la plate-forme des services pour leur traitement. Il convient de mentionner qu'avant

l'ouverture de la session de services, il nous faut ouvrir les autres sessions (d'accès et réseaux).

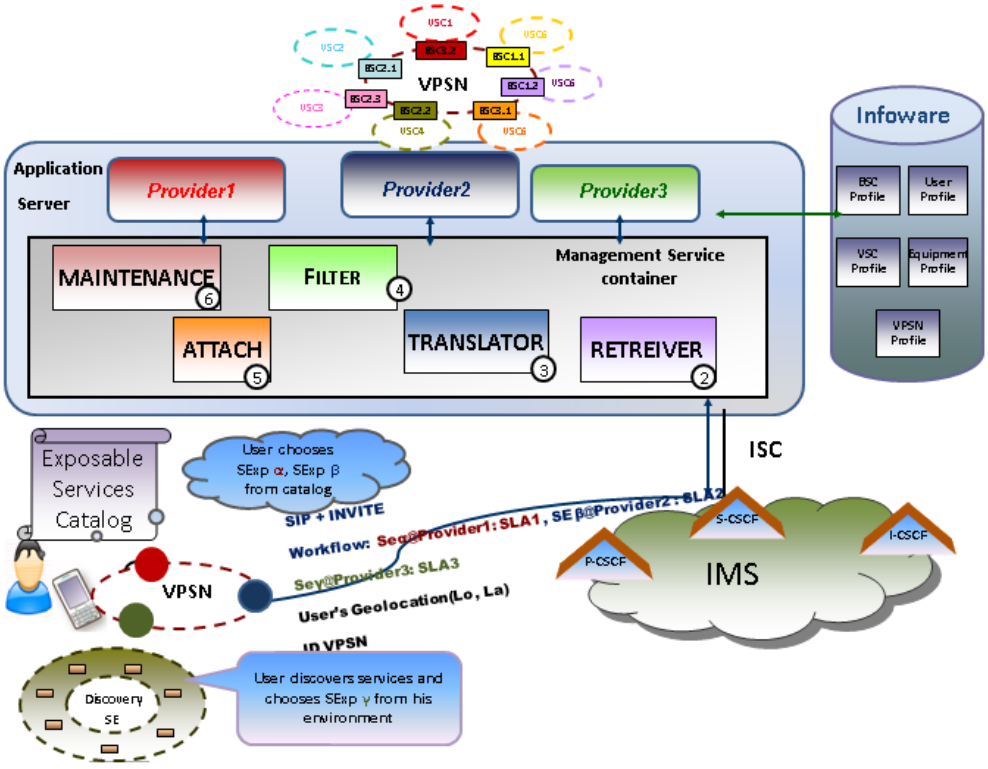

Figure 25: Session de service UBIS

(b.2) Initialisation de la session de services: SIP + INVITE

Au moment de l'initialisation de la session de service (Figure 26), la phase de négociation de la qualité de service de bout en bout est responsable de l'identification des composants de service impliqués dans le service global à rendre. Lorsqu'un l'utilisateur initie sa session, un processus de négociation de la QoS est déclenché pour identifier les composants de service selon le niveau de QoS qu'ils sont capables de fournir. Cette négociation dynamique de la QoS s'appuie sur la méthode INVITE du protocole de signalisation SIP+, qui fournit une description de QoS dans un niveau plus élevé couvrant le service global demandé par l'utilisateur. Ce protocole permet une déclinaison Top-Down de la négociation QoS du niveau service (QoS du composant de service) jusqu'au niveau des équipements (QoS des équipements) en passant par le niveau réseau (La QoS des réseaux d'accès et de transport).

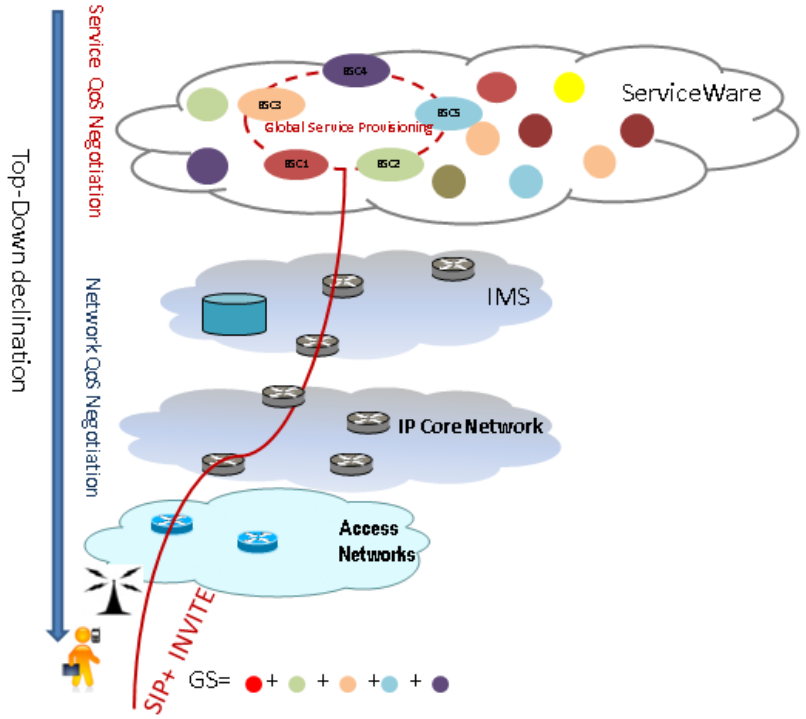

Figure 26: SIP+ INVITE pour une Négociation de la QoS de bout en bout

Le traitement de la demande d'initialisation d'une session de services est lancé par la plate-forme de services suite à la réception du message SIP + INVITE, qui contient toutes les informations concernant la position géographique de l'utilisateur (Lo, La), et également celles relatives à la logique de la composition des services exposables, qui sont sélectionnés par l'utilisateur au moment de la phase de personnalisation des services, et qui vont éventuellement participer à la session «user -centric». Ces informations sont introduites dans le «Body» du message SIP+ INVITE (Figure 27).

Le corps du message SIP+ INVITE contient donc principalement la logique des services, elle définit la liste des transactions des services exposables et leur ordre d'exécution au cours de la session user-centric {SExpα @provider1, SExpβ @ provider2, SExpγ @ provider3}. Il contient également une description de la QoS (SLA) de chaque SExp demandé par l'utilisateur. En particulier, les exigences de l'utilisateur sont décrites pour chaque SExp en fonction du modèle QoS basé sur les quatre critères de QoS (Disponibilité, délai, fiabilité et capacité).

A la réception de ce message, le ServiceWare, va déclencher les mécanismes de négociation de la QoS pour traduire les services exposables en services de base «BSC (Basic Services component)» avec la QoS requise, et va compléter le VPSN.

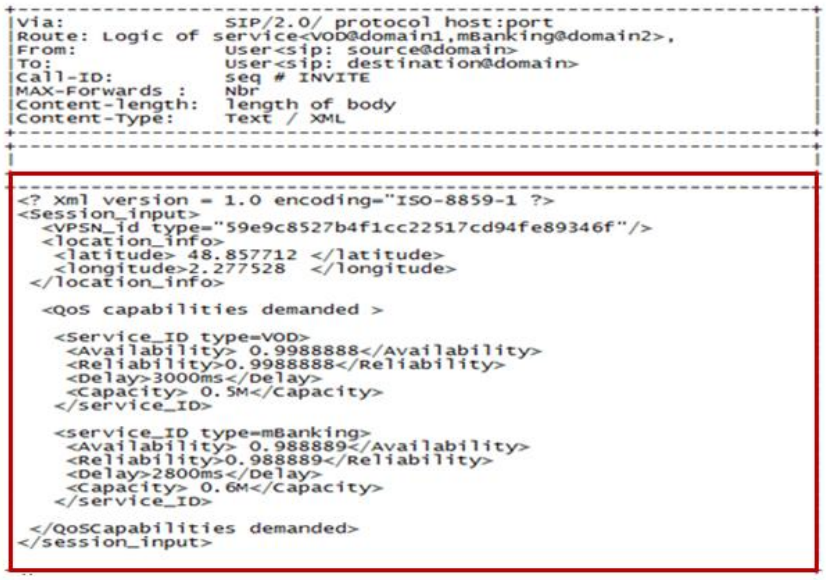

Figure 27: Structure du message SIP+ INVITE

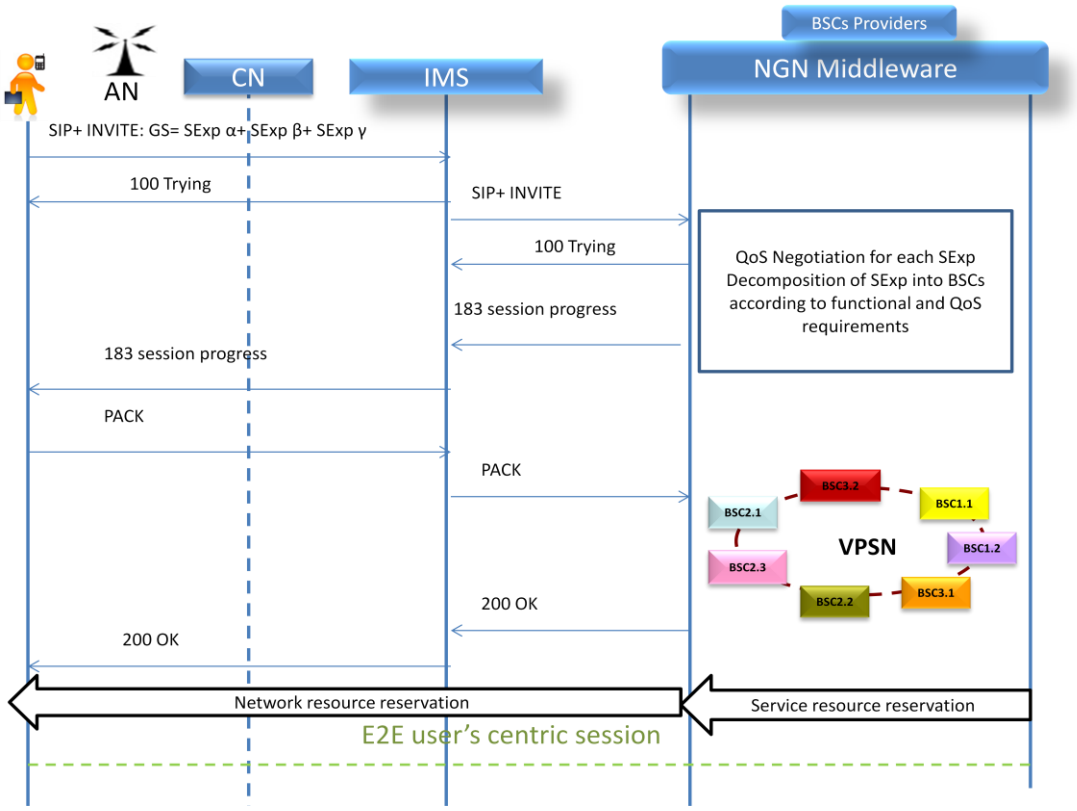

Figure 28: Diagramme de séquence (Initialisation de la session de service)

(b.3) Processus de la Création du VPSN :

Pour atteindre une gestion et un contrôle automatisés et distribués de la qualité de service au niveau service de l'architecture, nous allons appliquer notre vision de SOA basée sur des composants de service génériques, mutualisables, Stateless, et autogérables. Un composant de service peut rendre un service de type usage, contrôle ou gestion.

Dans notre approche, chaque service exposable est composé d'un ensemble de composants de service de base « BSC» de type usage. En outre, la session de service est créée en définissant la chaine logique des composants de services exécutables de type usage « BSC». Cette chaine logique constitue le VPSN (Virtual Private Service Network). Ces BSCs seront classés selon les transactions définies par l'utilisateur au moment de l'ouverture de la session de service.

Le scénario que nous allons décrire se base sur une configuration où nous n'avons qu'une seule plate-forme de service. Dans le cas où nous aurions plusieurs plates-formes de service

(le terminal pourrait en être une), la création du VPSN est partagée, chacune des plates-formes le complétant avec les services dont elle a la responsabilité.

La composition de service permet l'interopérabilité, l'interaction et la coopération entre différentes capacités des composants de service c.-à-d. l'aspect fonctionnel et non fonctionnel du service pour enrichir le service pour l'utilisateur.

Pour créer le VPSN en se basant sur les informations envoyées via le SIP+ INVITE, on utilise les composants de service de type gestion «Management Service Components: MSC» afin d'assurer la création et la maintenance du VPSN dans la plate-forme de service. Notre conception est basée sur un ensemble d'interactions entre des MSC avec des interfaces en couplage lâche facilitant l'interaction et l'interopérabilité pour créer et maintenir le profil VPSN. En effet, le processus de création est basé sur cinq MSCs (**Erreur ! Source du renvoi introuvable.**Figure 29).

- Retriever_SIP «RMSC»: extrait les informations contenues dans le Body SIP+ INVITE
- Translator «TMSC»: traduit le workflow des services exposables et le SLA en composants de service exécutables avec la QoS demandée. On mentionne que la combinaison des fonctions des composants de service et leur QoS permet d'assurer les besoins des utilisateurs.
- Filter «FMSC»: trouve la combinaison (ID-BSC, @ logique) selon la liste des ID-BSC limitée par un cercle avec un rayon Я et un centre {LO, LA} qui correspond à la position géographiques de l'utilisateur au moment de l'établissement de la session pour offrir des services selon le contexte ambiant de l'utilisateur.
- Attach_VPSN «AMSC»: ajoute l'ID-VPSN au profil du composant de service. Un composant de service peut être mutualisé entre plusieurs VPSN. Donc l'AMSC vérifie la QoS statistique en se basant sur les valeurs de conception pour contrôler l'attachement d'un composant de service à un nouveau VPSN.
- Maintenance_VPSN «MMSC»: ajoute les informations du composant de service (ID-BSC, @ logique au profil VPSN. Durant la session de service, le MMSC maintient le VPSN en mettant à jour le profil VPSN dans le cas où la QoS est dégradée. Il assure le

remplacement des informations du composant de service dégradé par celle d'un autre composant qui est fonctionnellement et QoS équivalent.

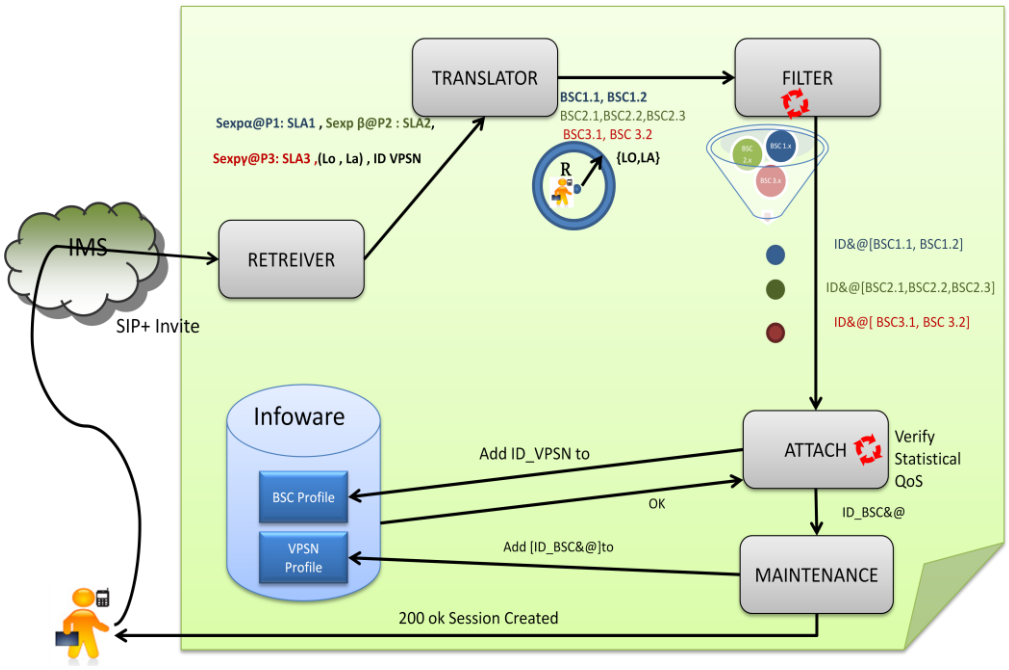

Figure 29: Processus de création de VPSN

La création du VPSN représente les différentes étapes pour créer le «profil VPSN». En effet, le profil VPSN contient la chaine logique de tous les composants de service BSC qui assurent la fonction des services exposables avec la QoS requise. Le processus de la création est lancé par la plate-forme de service après avoir reçu le message SIP+ INVITE à travers IMS.

Premièrement, le RMSC extrait les informations à partir du corps du message SIP+ INVITE. Deuxièmement, le TMSC est invoqué pour traduire le workflow des services exposables en une liste de BSC exécutables avec les critères de QoS requis: {SExpα=BSC1.1+BSC1.2, SExpβ=BSC2.1+BSC2.2+BSC2.3, SExpγ =BSC3.1}. Le résultat est la liste des BSCs avec la QoS offerte adéquate.

Troisièmement, le FMSC est sollicité pour trouver la combinaison {ID-BSC, @logique} pour chaque BSC selon la position géographique de l'utilisateur pour gagner en terme de temps de réponse durant la phase opérationnelle, parce qu'il existe plusieurs BSC ubiquitaires qui sont déployés dans différentes plates-formes.

Initialement, la recherche est effectuée dans la zone ambiante de l'utilisateur qui est limitée par un rayon R et le résultat est envoyé par la suite à l'AMSC. Si la recherche n'aboutit pas, on incrémente le rayon R jusqu'à ce qu'on trouve le BSC qui réponde au besoin de l'utilisateur.

Ensuite l'AMSC utilise le résultat obtenu pour vérifier les capacités statistiques de chaque BSC de la liste pour joindre le VPSN. Si le résultat est positif, l'AMSC ajoute l'ID-VPSN au profil du BSC dans la base de connaissance «INFOWARE».

Ensuite, l'infoware envoie un évènement au MMSC pour ajouter le BSC sélectionné à la table VPSN. Finalement, la table VPSN est créée et elle contient la chaine logique de tous les BSC et leur ordre d'exécution. Le diagramme de Séquence (Figure 28) représente les différentes interactions entre l'utilisateur, les composants de service de type gestion et l'infoware.

A la fin de ce processus nous avons pré-provisionné nos composants de service par le double attachement (BSC/VPSN et VPSN/BSC) sans réserver effectivement les ressources. Le provisioning se fait au moment de la consommation des services. La consommation des services au cours de la session « user-centric » se fait en fonction des transactions définies par l'utilisateur au moment de l'ouverture de la session, une transaction permet l'exécution d'une sélection d'un ensemble de services exposables. La consommation de service se traduit par la réservation réelle des ressources (provisioning) pendant la phase de l'exploitation et plus précisément au moment de la consommation effective.

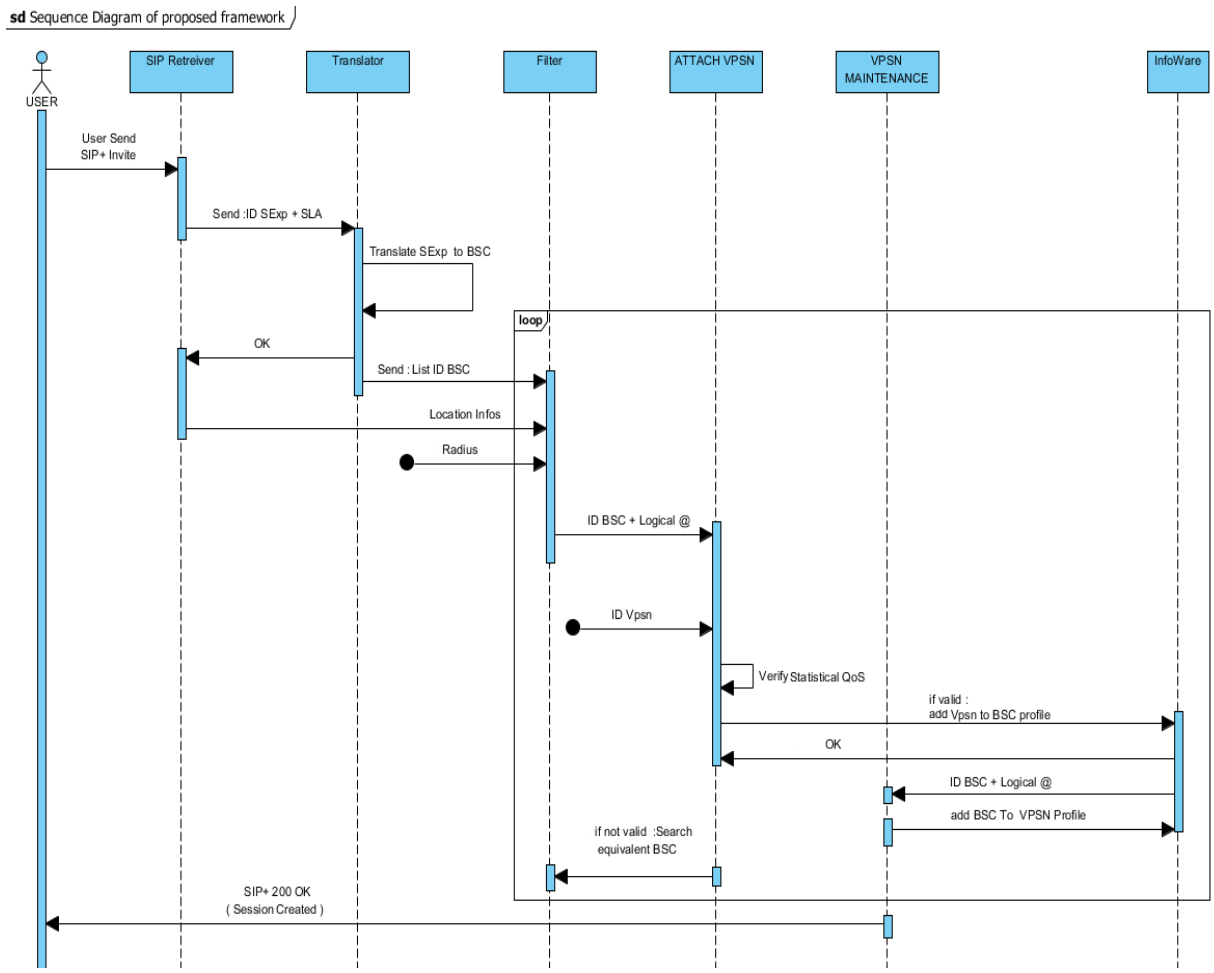

Figure 30: Diagramme de séquence pour la création du VPSN

(b.4) La modification de la session UBIS

Pour offrir à l'utilisateur une continuité de service avec une QoS de bout en bout dans un contexte hétérogène et mobile, il faut lui garantir une session unique et continue pendant la phase d'exploitation.

En fait, le bout en bout suppose que, hormis les ressources réseaux, les ressources services devraient être prises en compte en tant que ressources de premier plan lors de la session de l'utilisateur. Sur la couche réseau, de nombreux protocoles sont utilisés pour déclencher le processus de renégociation de la QoS si une dégradation de la QoS réseau se produit. Par contre ce problème n'est pas encore étudié dans la couche de services. Maintenir, donc, une session de service unique à l'utilisateur nécessite de fortes interactions plus flexibles qui

agissent sur la couche de service pour renégocier la QoS. Un événement de renégociation de la QoS au niveau service peut être déclenché si une dégradation de la QoS d'un composant de service se produit suite à une insuffisante de ressources ou bien elle peut également être invoquée par l'utilisateur qui décide volontairement de changer de services selon ses préférences, comme le passage d'une vidéo de son téléphone PDA à son PC.

a) Le protocole SIP+: Message SIP+ NOTIFY

Avec la convergence du plan de contrôle et du plan de gestion, le protocole SIP+ au niveau service peut transporter en plus des informations de contrôle les informations de gestion pour garantir plus de dynamicité pendant la phase d'exploitation. En effet, cette « QoS signalisation » communiquera en plus de la description des médias, les exigences de QoS au niveau des composants de service et elle enverra la notification de l'état de la QoS surveillée (in/out contrat) par le message SIP+ Notify d'un BSC faisant partie d'un VPSN à sa communauté VSC. Le protocole SIP+ offre une signalisation dynamique au niveau service pour fournir le service demandé et garantir la continuité de session en conformité aux SLA.

La Figure 31 représente la structure d'un message SIP+ NOTIFY utilisé pour gérer la session de service. Les champs utiles dans l'en-tête du message SIP+ NOTIFY qui sont utilisés pour notifier l'état de QoS (IN / OUT contrat) d'un composant de service en cours d'exécution sont les suivants:

- Allow-Events: il permet la notification des événements concernant le contrat de qualité de service (IN/OUT contract).
- Event: Ce champ est utilisé pour envoyer l'état de qualité de service (IN Contract/Out Contract) d'un composant de service.

```
+-----------------------------------------------------+
|Via:                 SIP/2.0/ protocol host:port     |
|From:                User<sip: source@domain>        |
|To:                  User<sip: destination@domain>   |
|Call-ID:             localid@host                    |
|Cseq:                seq#NOTIFY                      |
|Content-length:      length of body                  |
|Content-Type:        Application/XML                 |
|Contact:             <sip:user@pc.example.com        |
|Subscription-State:  Active                          |
|Allow-Events :       contract(IN/OUT contract)       |
|Event:               IN/OUT contract                 |
+-----------------------------------------------------+
+-----------------------------------------------------+
|                                                     |
+-----------------------------------------------------+
+-----------------------------------------------------+
| <?xml version="1.0" encoding="UTF-8"?>              |
| <contract-response version="1.0"  text="IN/OUT" />  |
|                                                     |
+-----------------------------------------------------+
```

Figure 31: Structure du message SIP+ NOTIFY

b) La communauté VSC

Au cours d'une session centrée sur les besoins de l'utilisateur « user centric », on peut avoir une dégradation de la qualité de service suite à un changement de la condition de QoS dans un nœud (par exemple la valeur courante dépasse les valeurs seuils). Le remplacement de ce dernier est géré par la communauté de services «VSC». La VSC regroupe les composants de service ubiquitaires qui sont fonctionnellement et QoS équivalents, ces communautés sont constituées et gérées par l'ensemble des fonctions des services de base (localisation, présence, découverte). Le pilotage de la qualité de service vise à évaluer l'état de la qualité de service de bout en bout au cours d'une session. Pour cela, il faut évaluer l'état de la qualité de service d'un composant de service qui intervient pendant la phase d'exploitation. La politique d'analyse utilisée repose sur un traitement interne de la QoS effectué par l'agent QoS et les notifications (In contrat / out contrat) reçues par la communauté VSC, pour assurer le remplacement d'un élément de service défaillant par un composant équivalent dans le cas d'une dégradation de la QoS.

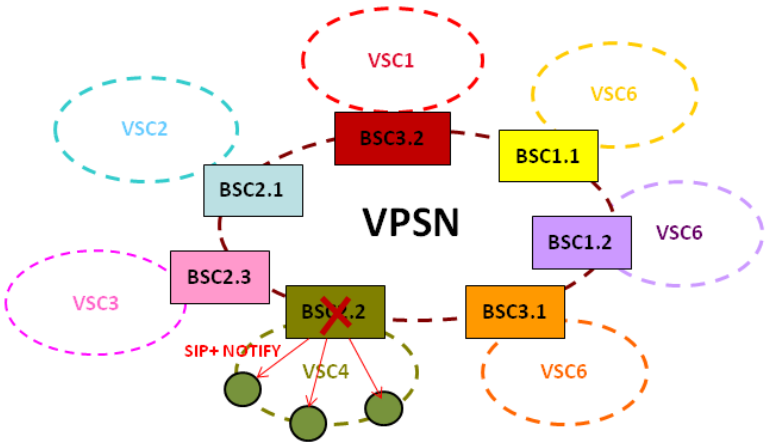

Figure 32: la communauté VSC

Pour cette raison, nous avons besoin d'identifier et de sélectionner l'élément le plus adéquat dans une communauté VSC en prenant en considération la QoS du nœud, du lien et du réseau car les composants de service peuvent être déployés dans des plates-formes hétérogènes, ce qui peut entrainer des temps de réponse variables. Pour faire face à cette hétérogénéité, nous proposons de s'appuyer sur les tables «QoS NLR» pendant la phase de sélection. En effet, chaque nœud de service dispose d'une table QoS qui contient les informations mises à jour de la QoS du nœud, du lien entre deux nœuds et aussi du réseau qui représente le réseau de transport entre ses nœuds. Dans la table VSC les composants de services sont classés selon la QoS du lien entre le BSC activé et ceux qui sont susceptibles de le remplacer.

- La QoS du Nœud représente les caractéristiques fonctionnelles de traitement responsables d'un processus spécifique.
- La QoS du Lien représente la qualité de service du canal de communication virtuel entre les deux nœuds.
- La QoS du Réseau définit la table de routage qui enregistre la qualité de service de chacun des chemins possible dans la couche de transport. La qualité de service est calculée et mise à jour en temps réel.

Pour ce faire, nous proposons une autogestion basée sur un algorithme «VSC QoS NLR selection» afin de trouver un autre composant de service avec une QoS équivalente pour le remplacer. Si cette autogestion de VSC échoue, la signalisation dynamique peut interagir avec le système de l'utilisateur dans la base de données, et réapprovisionner la qualité de service

dans le réseau de services (VPSN) afin de trouver un autre composant de service de remplacement avec une QoS correspondante aux préférences de l'utilisateur.

Au niveau du VSC, la communication se fera en P2P. Dans ce cas, la gestion de la QoS est répartie. En effet, l'intelligence est distribuée sur les composants de service du VSC. En d'autre terme, la requête SIP+ QoS Notify peut être envoyée aux BSCs du VSC en P2P selon deux approches:

- ***1ère approche: Transmission P2P de proche en proche***

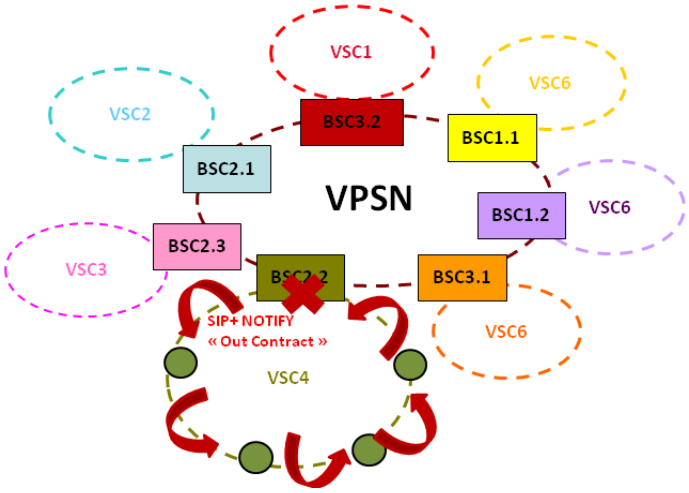

Figure 33:VSC Transmission P2P de proche en proche

La requête SIP+ QoS Notify est envoyée de proche en proche selon le classement QoS NLR des éléments de service dans la table «VSC QOS NLR». La sélection du candidat s'appuie sur l'algorithme «QoS NLR selection» (Figure 34) pour élire le composant qui offre une QoS NLR équivalente ou bien meilleure de celle du composant qui est hors contrat. Cet algorithme s'appuie sur les tables VSC QOS NLR, pour que la requête SIP+ Notify soit envoyée en premier lieu vers un composant de service qui a probablement les ressources nœud, lien et réseau pour satisfaire la demande.

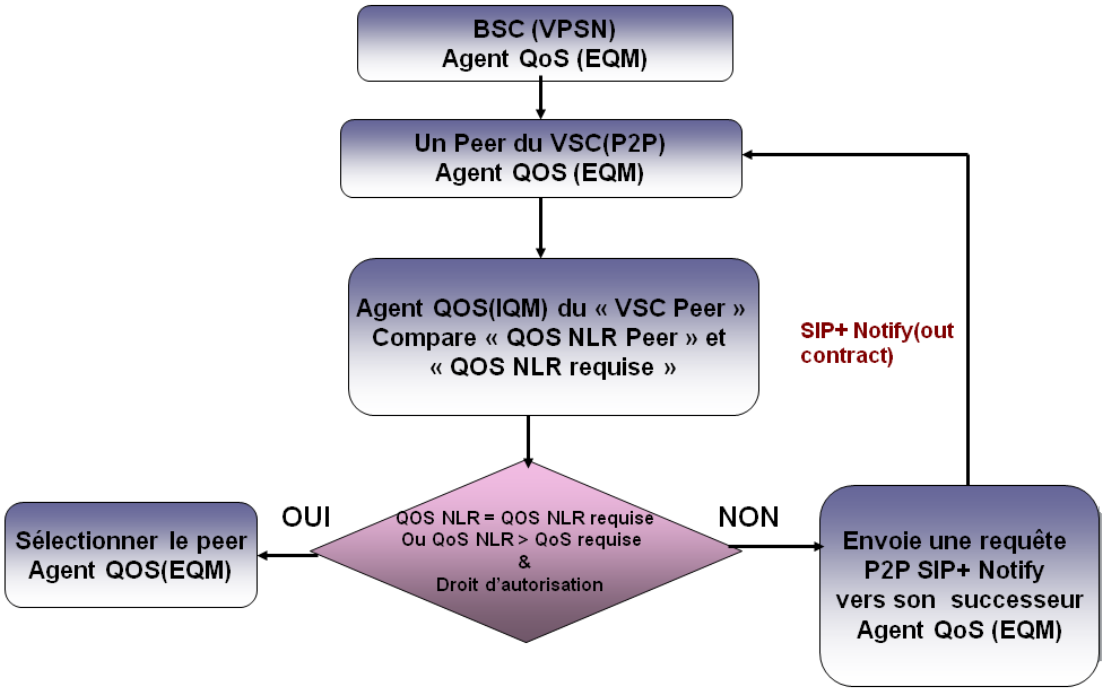

Figure 34: Algorithme «QoS NLR selection» : proche en proche transmission

Lorsque l'agent QOS (IQM) du BSC (VPSN) détecte une dégradation de la QoS, l'EQM assure la notification de la défaillance de ce dernier via le protocole SIP+ Notify (QoS NLR requise) aux BSC de la communauté VSC, et cela, pour assurer son remplacement par un autre BSC offrant une QoS NLR équivalente à celle du composant défaillant. Ce calcul se fait par une communication en P2P SIP+, le P2P SIP+ permet d'étendre le protocole SIP+ au niveau du VSC et le router selon le mécanisme P2P.

Le VSC constitue le réseau des BSCs en P2P, la requête SIP+ Notify est envoyée de proche en proche selon l'ordre recommandé par la table QoS NLR, qui favorise le BSC qui a une QoS équivalente ensuite les BSCs qui ont une «QoS NLR» meilleure.

Le premier BSC de ce réseau traite la requête SIP+ Notify. Si ce dernier a les droits, il envoie une requête P2P «SIP + message SET» pour rejoindre le VPSN et par la suite remplacer le nœud qui est hors contrat, sinon il assure le routage de cette requête à son successeur de la communauté VSC en s'appuyant sur la table QoS NLR.

Les différentes possibilités de traitement:

- Si le BSC a les droits d'autorisation et la QoS NLR BSC_i «VSC» >= QoS NLR Conception du BSC défaillant, ce nœud envoie un message SIP+ MESSAGE SET au VPSN pour assurer le remplacement de ce dernier.

- Si le BSC n'a pas les droits d'autorisation, ce nœud renvoie un message SIP+ Notify (out contrat) à son successeur selon la logique de la table QoS NLR(VSC) jusqu'à la satisfaction de la demande.

  - ***2ème approche: Transmission P2P multicast***

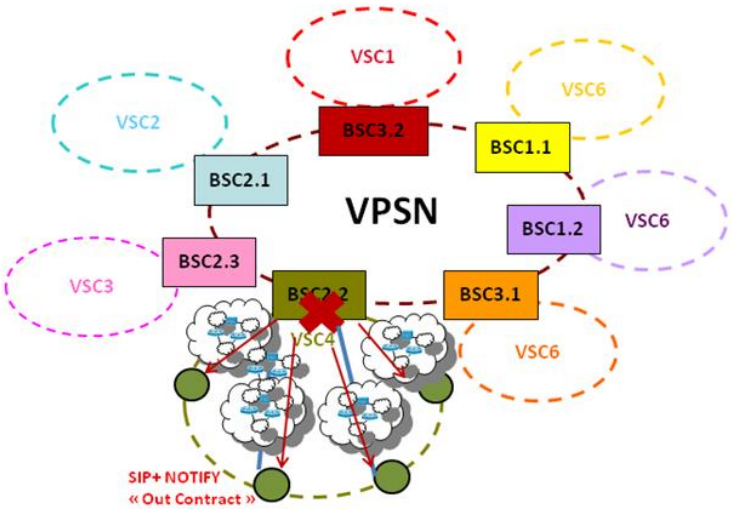

Figure 35: Transmission P2P dans la communauté VSC en multicast

Dans cette approche, le SIP+ QoS Notify est envoyé en multicast à tous les éléments de service de la communauté P2P. Dans ce cas la sélection du candidat se fait selon la table QoS NLR. En effet, dans la table QoS NLR les BSCs sont classés suivant leur QoS NLR par rapport au BSC activé. Le BSC classé au début de la table aura forcément une QoS équivalente ou meilleure à celle du BSC hors contrat. S'il est autorisé, il envoie un message SIP+ message SET pour rejoindre le VPSN.

Dans le cas ou plusieurs BSC dans la table ont une QoS NLR équivalente à celui qui est hors contrat, le BSC qui a un meilleur temps de réponse sera sélectionné pour rejoindre le VPSN.

Le diagramme (Figure 36) représente le déroulement de l'algorithme QOS NLR Selection.

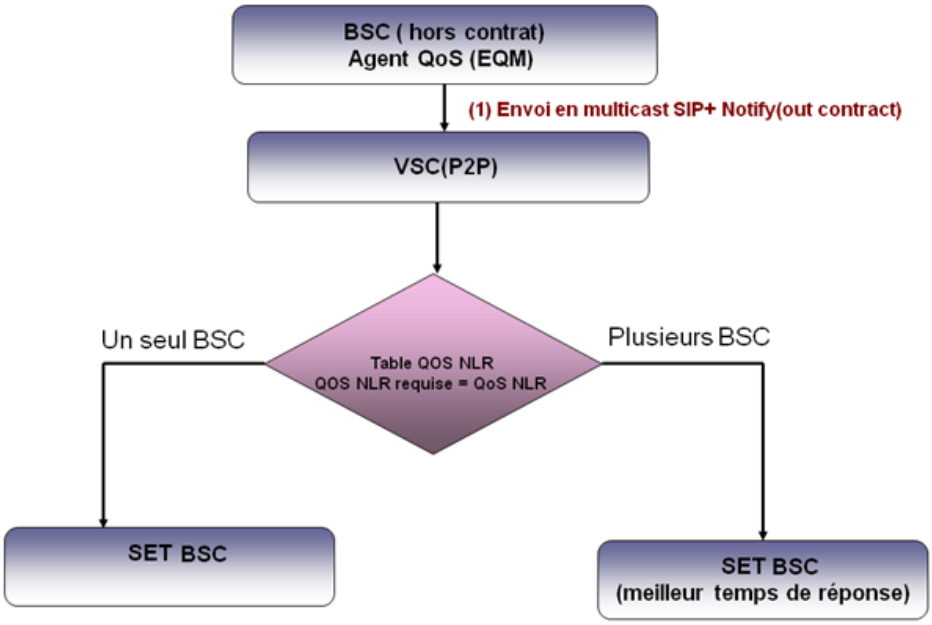

Figure 36: Algorithme «QoS NLR Selection»: transmission P2P en multicast

La Figure 37 représente les différents messages échangés entre le BSC défaillant du VPSN et la communauté du VSC. Au début de la session, la requête Subscribe Event QoS est envoyée en SIP+ P2P par tous les éléments de service de la communauté VSC aux BSCs (VPSN) pour qu'ils souscrivent à l'événement (Out contrat) afin d'être notifiés de son occurrence.

Le BSC (VPSN) répond par un message 200 OK, ce message indique que la demande de souscription a été bien reçue.

Les agents QoS(EQM) des BSCs appartenant à la communauté VSC restent à l'écoute. Si une dégradation de la qualité de service de BSC (VPSN) se produit pendant la phase d'exploitation, l'agent QOS (EQM) de ce dernier envoie un message de notification SIP+ à la communauté VSC. Ce message est routé en P2P et traité selon l'algorithme «QoS NLR Selection» jusqu'à la sélection du peer de remplacement.

Lorsqu'un agent QoS du VSC se reconnaît candidat de remplacement en s'appuyant sur l'algorithme QoS NLR Selection, il envoie un message SIP+ MESSAGE SET pour rejoindre le VPSN.

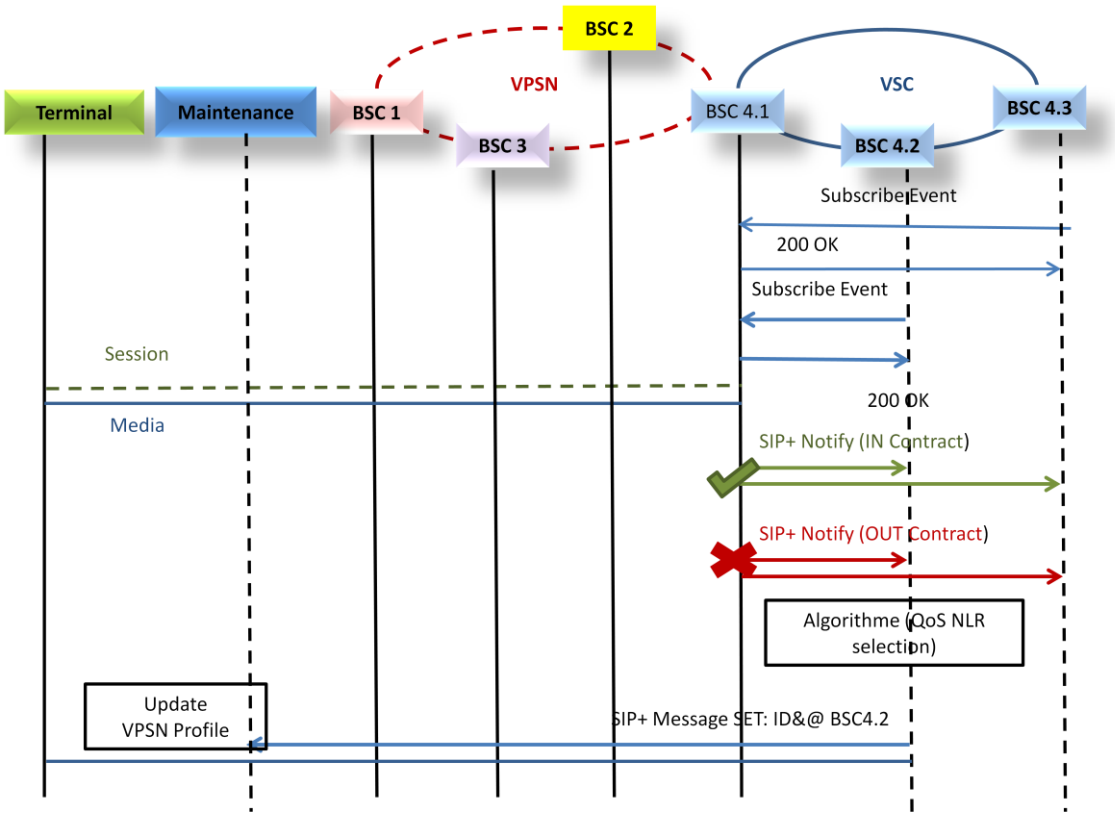

Figure 37: Diagramme d'autogestion du VSC

Nous décrivons un scénario (Figure 38) qui illustre l'utilité de l'automatisation de la gestion de la session de l'utilisateur pendant les différents types de mobilité. Lorsque l'utilisateur (Point A) est à la maison (localisation {LO, LA}), il choisit une composition de services exposables : SExpα, SExp β et SExp γ (**Erreur ! Source du renvoi introuvable.**Figure 23), donc après traitement de cette requête par les composants de gestion du ServiceWare, la logique des composants de service de base selon les préférences de l'utilisateur (pré-provisioning) est BSC1.1 + BSC1.2 + BSC2.1 + BSC2.2 + BSC2.3 + BSC3.1. Durant la phase d'exploitation, la performance du BSC2.2 (Point B) se dégrade, et elle ne peut pas continuer à fournir la QoS requise par le SExpβ. Par conséquent, le BSC2.2 envoie un message SIP+ Notify à tous les membres de sa communauté VSC pour procéder à son changement. Il existe toujours des services ubiquitaires qui ont les capacités fonctionnelles et non-fonctionnelles (QoS) pour remplacer le composant de service défaillant, l'algorithme "QoS NLR Selection" est utilisé pour faire le choix dans la communauté VSC, en se basant sur la localisation de l'utilisateur

pour garantir un temps de réponse équivalent au temps de réponse fourni par le composant dégradé.

Lorsque la communauté VSC sélectionne le composant BSC2.2' (point B') localisé dans une autre plate-forme de services. Le gestionnaire de la VSC invoque le composant de gestion qui est responsable du maintien du VPSN «MMSC» en envoyant un message qui contient les informations sur le BSC2.2' du remplacement (ID-BSC2.2', @logique), ID-VPSN) pour mettre à jour le profil VPSN. Ensuite la connexion est transférée du BSC2.2 au BSC 2.2' (provisioning dynamique).

L'utilisateur (point A) se déplace (mobilité du terminal), et quand il arrive au bureau (point A'), il change son terminal du PDA à un PC (mobilité de l'utilisateur). Dans cette situation, en raison du changement de l'emplacement de l'utilisateur du point A {LO, LA} au point A{LO', LA'} et le changement de son terminal du PDA au PC, les composants de service de base BSC1.1+ BSC1.2 ne peuvent plus répondre à ses exigences. Pour cela, le composant de maintenance de VPSN se charge de les remplacer par deux nouveaux services BSC4.1 + BSC4.2, qui sont adaptés à son nouveau terminal. Par conséquent, nous obtenons un réapprovisionnement dynamique du VPSN après la mobilité: BSC2.1 + BSC2.2' + BSC2.3 + BSC3.1+ BSC4.1+ BSC4.2.

Finalement, la session de service de l'utilisateur est maintenue et le service delivery est assuré. Nous mentionnons que toutes ces opérations sont transparentes à l'utilisateur et elles sont effectuées d'une manière autonomique durant la session.

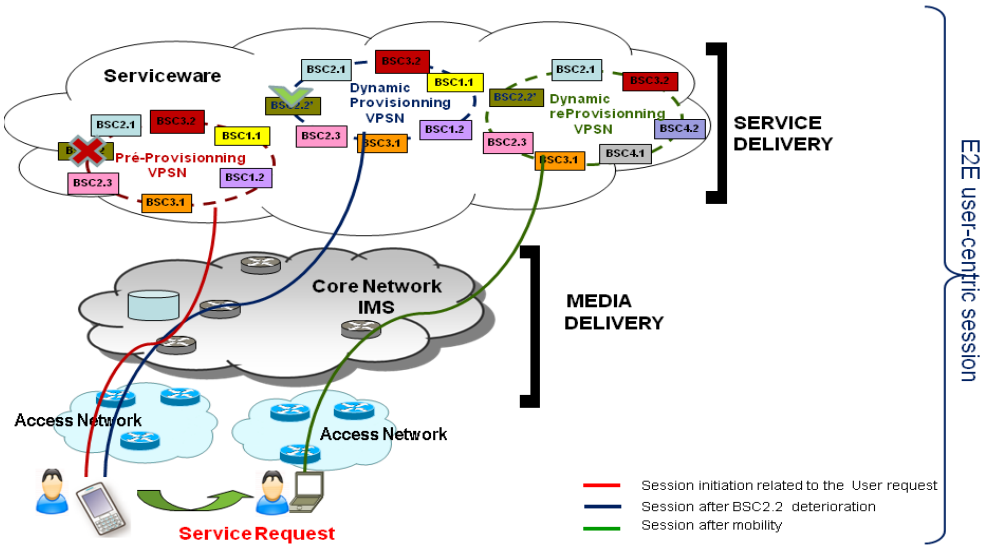

Figure 38: Scénario d'autogestion pendant la mobilité

V.4 Conclusions des propositions

Dans ce chapitre, nous avons consigné nos propositions qui visent à répondre au besoin du pilotage dynamique de la qualité de service de bout en bout d'une session «User-centric» qui tient compte des besoins fonctionnels et non-fonctionnels (QoS) de l'utilisateur et de ses préférences dans son nouveau contexte qui se caractérise par l'hétérogénéité et la mobilité.

Pour faciliter ce pilotage avec un maximum de dynamicité et de flexibilité, nous avons centré notre étude sur 3 axes principaux: la dimension organisationnelle, la dimension fonctionnelle (le composant de service autonomique) et la dimension protocolaire.

Basée sur les travaux existants de notre groupe de travail qui traitent la dimension architecturale représenté par l'architecture UBIS et la dimension informationnelle représentée par le modèle informationnel, nous avons proposé une nouvelle organisation pour faire face au problème de gestion dans les architectures contemporaines, et avoir une gestion distribuée qui favorise le principe de coopération entre les acteurs de l'architecture selon leur rôle pour répondre à un besoin non-fonctionnel (QoS) qui sera piloté à tous niveaux architecturaux.

Dans l'objectif de satisfaire le SLA contacté, Nous avons proposé un composant de service autonomique qui dispose d'un agent de QoS qui contrôle ses ressources en temps réel et qui réagit dynamiquement et de manière autonome dans le cas d'un changement dans le contrat de QoS (Disponibilité, Fiabilité, Délai et Capacité) au cours de la session de l'utilisateur. Le contrôle et la gestion de l'agent QoS s'appliquent durant toutes les phases de la session (pré-provisioning, provisioning et consommation) pour garantir à l'utilisateur une QoS en totale adéquation avec ses attentes et le prix qu'il paye.

Nous avons proposé dans la dimension protocolaire, un protocole de signalisation SIP+ qui a pour rôle l'établissement, la modification et la terminaison d'une session de service unique et continue dans une architecture trans-organisationnelle. Ce protocole sert à une négociation de la QoS des services personnalisés lors de la phase d'initialisation de la session de service et aussi à une renégociation de la QoS lors de la phase d'exploitation, pour remédier aux différents problèmes de dégradation de la QoS qui sont principalement causés par la mobilité.