Etude de Problème de Sélection de Services Web en tenant Compte des contraintes temps réel

TURKI KOSSENTINI[1]Hazar, Dr. Leila BACCOUCHE[2],
Pr. Henda HAJJAMI BEN GHEZALA[3]
Laboratoire RIADI-GDL, Ecole Nationale des Sciences
de l'Informatique, Campus Universitaire de la Manouba, 2010 Manouba,
Tunisie
[1] Hazar.Turki@riadi.rnu.tn
[2] leila.baccouche@insat.rnu.tn
[3] Henda.BG@cck.rnu.tn

Résumé : Lorsqu'une application est de type web, ses tâches composantes peuvent être exécutées à l'aide des services web. En particulier, pour une même tâche, on peut découvrir plusieurs services web aptes à l'exécuter, on serait alors face à un problème de sélection de services web afin de choisir la combinaison de services qui offre la meilleure qualité de service. Cette sélection devra aussi prendre en considération le respect des contraintes temps réel imposées lorsqu'il s'agit d'une application temps réel. Dans ce papier, nous étudions d'une manière générale le problème de sélection de services web basée sur la qualité de service. Puis, nous étudions la classification des critères de qualité de service afin d'exprimer l'aspect temporel des services web en recensant les critères de qualité de service optimisant le temps. Enfin nous présentons une étude de cas pour une application temps réel dans laquelle nous sélectionnons les services web qui respectent les contraintes temporelles imposées.

Mots clés : Services web, contraintes temps réel, sélection de services web, critères de qualité de service.

Introduction

Le temps joue un rôle de plus en plus important dans les services et les applications actuelles (informatique industrielle, télécommunications, multimédia, ...). Cependant parmi les applications qui tiennent compte de l'aspect temps, nous citons les applications temps réel qui se distinguent des applications traditionnelles par les contraintes temporelles qu'elles doivent respecter et qui sont généralement exprimées sous forme d'échéances et de durées de validité.

Par ailleurs, une application temps réel peut être de type web et les tâches qui sont composantes de cette application peuvent être exécutées par des services web. En conséquence, ces services web doivent respecter des contraintes temporelles qui représentent la contrainte fondamentale d'une application temps réel.

Les services web sont des systèmes logiciels qui répondent généralement à des besoins fonctionnels. Les contraintes temps réel expriment des exigences en termes de temps sur le comportement de l'application. Ces exigences sont totalement indépendantes des besoins fonctionnels de l'application. Ainsi les contraintes temps réel seront exprimées à travers des besoins non fonctionnels. Les besoins non fonctionnels des services web s'expriment généralement à travers la qualité de service (QdS).

La QdS est un ensemble de propriétés et caractéristiques d'une entité ou d'un service qui lui confèrent l'aptitude à satisfaire des besoins déclarés ou implicites [2]. Ces besoins peuvent être liés à des paramètres tels que l'accessibilité, la disponibilité, le temps de réponse, le coût, la fiabilité, etc. Ces paramètres peuvent être alors considérés comme un critère de choix lorsqu'on a affaire à sélectionner parmi plusieurs services web découverts ceux qui respectent les contraintes de temps imposées.

En s'accordant avec ces visions, nous étudions d'une manière générale le processus de sélection des services web dans la section 2. Dans la section 3, nous présentons comment prendre en considération les contraintes temporelles lors d'une sélection des services web exécutant une application temps réel. Pour mettre en évidence le respect des contraintes de temps lors d'une sélection des services web, nous proposons dans la section 4 une étude de cas qui présente une démarche à suivre pour exécuter une application temps réel à l'aide des services web. La dernière section conclut l'article.

1. Sélection de services web basée sur la qualité de service

La sélection consiste à choisir, parmi les services web découverts, ceux qui répondent au mieux aux exigences de l'utilisateur sur la base des besoins fonctionnels et/ou non fonctionnels. Les besoins non fonctionnels des services web sont généralement exprimés à l'aide des critères de QdS. Dans une sélection de services web basée sur la QdS deux cas se présentent [3] :

Si la réponse à la requête d'un client exige la sélection d'un seul service web non composite, alors la sélection est très simple. Le candidat qui présente la meilleure QdS sera désigné pour répondre à la demande du client.

Si la réponse à la requête d'un client exige la combinaison de plusieurs services existants, alors la sélection dans ce cas sera plus complexe du fait qu'il faut choisir la combinaison des services composants qui répond mieux aux besoins des clients.

Pour le deuxième cas nous sommes face à un problème de sélection de services web composants basée sur la QdS. La modélisation de ce problème dépend de la nature de la stratégie de sélection : locale ou globale.

1.1. Stratégies de sélection de services web

L'exécution d'un service web composite inclut nécessairement une étape de sélection de services web composants. La manière de sélectionner ces services web dépend des exigences de choix imposées. Dans un contexte temps réel, ces exigences peuvent être des contraintes temporelles.

Dans la littérature, il existe deux stratégies de sélection de services web [13] : une stratégie de sélection locale et une stratégie de sélection globale. Chaque stratégie est définie en fonction de la nature des contraintes de QdS imposées. En fait, ces contraintes n'ont pas un caractère obligatoire, mais permettent de définir les limites de fonctionnement d'un service web composant ou composite.

La stratégie de sélection locale [1, 3 et 4] a pour objectif de choisir le meilleur service web pour chaque tâche individuelle à part entière en considérant des contraintes de QdS relatives à chaque tâche plutôt qu'en considérant des contraintes de QdS globales exprimées pour l'ensemble des tâches. Pour le cas d'une application temps réel, il s'agit de sélectionner pour chaque tâche un service web apte à l'exécuter en prenant en compte les contraintes temps réel locales et d'autres préférences imposées pour chaque tâche. Par exemple, la durée d'exécution d'une tâche devra avoir un temps de réponse qui ne dépasse pas quelques minutes, le service web exécutant une tâche doit être disponible à 100%, etc.

La stratégie de sélection globale [10 et 12] a pour but de choisir la combinaison de services web qui garantit la meilleure qualité globale en tenant compte des contraintes de QdS et des préférences globales assignées pour l'ensemble des tâches. Pour le cas d'une application temps réel, il s'agit de sélectionner la combinaison de services qui offre la meilleure QdS et qui respecte les contraintes temps réel et d'autres préférences globales imposées pour l'ensemble des tâches de l'application comme par exemple imposer que la durée d'exécution de bout en bout doit être inférieure à une échéance.

Appliquer l'une ou l'autre des stratégies revient à dérouler un algorithme de sélection approprié. Par exemple, dans [3] « Jaeger et Muhl » proposent un algorithme de sélection qui applique le principe de la stratégie de sélection locale et qui s'inspire de l'algorithme Glouton [5, 7 et 11] proposent d'appliquer des algorithmes de parcours de graphes, un algorithme de recherche exhaustive et un algorithme de programmation dynamique afin d'appliquer le principe de la stratégie de sélection globale.

1.2. Modélisation du problème de sélection de services web

La modélisation du problème de sélection dépend de la nature de la stratégie de sélection appliquée : locale ou globale.

Pour la stratégie locale, la modélisation du problème de sélection est destinée aux services web composants dont l'enchainement est défini dès l'étape de conception. Dans [3 et 4], « Jaeger et Muhl » et « Liu, Ngu et al. » appliquent la technique SAW (Simple Additive Weighting) pour modéliser le problème de sélection de services web basée sur la QdS. Etant donné un ensemble de services ayant la même fonctionnalité mais des valeurs de critères de QdS différents, le principe de la technique SAW consiste à affecter un vecteur de QdS à chaque service web candidat dans la sélection. Ce vecteur contient les valeurs des critères de QdS qui caractérisent chaque service web. Sur la base de ce vecteur, il s'agit d'appliquer les deux phases suivantes :

- Phase d'ajustement : Elle consiste à ajuster les critères de qualité de dimension croissante (comme la disponibilité) et décroissante (comme le temps de réponse) inclus dans chaque vecteur en appliquant des équations de normalisation afin de rendre les valeurs des critères de QdS homogènes.

- Phase d'attribution de poids : Lors de cette phase, il s'agit d'attribuer un poids à chaque critère de QdS inclus dans le vecteur de QdS puis de calculer le score de chaque service web. Ce score représente une somme pondérée des valeurs

normalisées de ces critères et le service web qui a le score le plus élevé est celui qui est attribué à la tâche en question.

Lorsqu'il s'agit d'appliquer la stratégie de sélection globale, le problème de sélection de services web composants peut être modélisé comme un problème d'optimisation combinatoire. Il s'agit, pour une composition fonctionnelle de services web et une QdS donnée, de trouver les services web qui sont disponibles et qui peuvent satisfaire une ou plusieurs contraintes de QdS tout en considérant comme objectif d'optimiser (maximiser ou minimiser) une fonction coût. Ainsi, lorsqu'il s'agit de considérer une seule contrainte de QdS, « Yu et Lin » [11] proposent de modéliser le problème de sélection comme un problème de sac à dos. Et lorsqu'il s'agit de considérer plus qu'une contrainte de QdS, « Yu et Lin » [10] et « Zeng, Benatallah et al. » [12] proposent de modéliser le problème de sélection comme un problème de programmation linéaire.

2. Prise en compte des contraintes temporelles à travers les critères de qualité de service

Les critères de QdS représentent l'aspect non fonctionnel d'un service web et c'est à l'aide de ces critères que les exigences non fonctionnelles du demandeur de services peuvent être exprimées. Ces différents critères peuvent être classifiés selon leur déterminisme. Ainsi un critère est déterministe lorsque sa valeur est connue ou certaine quand le service web est invoqué (exemple : prix d'exécution, taux de pénalité) et il est non déterministe lorsque sa valeur est incertaine lorsque le service web est invoqué (exemple : durée d'exécution).

Par ailleurs, les critères de qualité peuvent être aussi classifiés en des critères génériques ou spécifiques au domaine. En effet, les critères génériques sont relatifs à tous les services web indépendamment des domaines auxquels ils appartiennent. Même si ces critères sont génériques, ils ne demeurent pas moins qu'ils sont classifiés de différentes manières selon les travaux de recherche [6 et 8].

Dans certains domaines, de tels critères génériques ne pourraient pas être suffisants. Le modèle de QdS devrait aussi inclure les critères spécifiques au domaine et être extensible. Ainsi dans un contexte temps réel, les critères spécifiques au domaine sont ceux relatifs au temps ou susceptibles de l'influencer. D'après une étude de la majorité des critères de QdS, nous avons jugé que les critères de QdS liés à la performance tels que le temps de réponse, la latence et le débit ainsi que les critères capacité et disponibilité sont des critères qui peuvent influencer le respect des contraintes temporelles imposées. Ces critères restent les plus adaptés pour décrire le

temps. Toutefois, d'autres critères temporels spécifiques aux applications temps réel peuvent être rajoutés.

Une fois les critères de qualité choisis, nous présentons dans la section suivante une étude de cas pour la mise en œuvre d'une application temps réel qui fait appel à des services web susceptibles d'exécuter ses tâches composantes.

3. Etude de cas pour la mise en œuvre d'une application temps réel basée sur les services web

Une application temps réel est composée d'un ensemble de tâches dont l'exécution doit respecter des contraintes temporelles explicitement spécifiées pour tenir compte de la dynamique de l'environnement physique des applications . Ces tâches peuvent être exécutées à l'aide des services web. Ces derniers doivent alors respecter les contraintes de temps imposées par l'application temps réel. Comme exemple d'application temps réel qui fait appel à des services web pour exécuter ses tâches, nous prenons celle de « réservation de circuits touristiques ». Dans les sections qui suivent nous décrivons en détail une architecture des services web adaptée aux applications temps réel, une description de l'application « réservation de circuits touristiques », une modélisation du problème de sélection des services web basée sur la QdS, un algorithme de sélection à appliquer et enfin une proposition des formules de calcul de la valeur des critères de QdS pour un service web composite.

3.1. Architecture des services web adaptée aux applications temps réel

Une application temps réel qui fait appel à des services web afin d'exécuter ses tâches devra passer par les processus de découverte, de sélection et de composition des services web pour affecter à la fin les services web qui sont aptes à exécuter chaque tâche et qui respectent les contraintes de temps imposées. Afin de gérer ces différents processus, nous avons utilisé une architecture de services web adaptée aux applications temps réel [9].

Comme le montre la figure 1, l'architecture utilisée est composée de quatre composants à savoir : l'annuaire UDDI étendu qui stocke des descriptions WSDL des services web publiés par différents fournisseurs ainsi que l'information de QdS de chaque service web, les fournisseurs qui offrent des services web aux clients, le client qui a besoin d'accéder à une application temps réel afin de lui rendre un service web composite exécutable qui répond à ses contraintes temps réel et ses préférences et enfin le courtier de QdS qui gère la découverte, la sélection et l'exécution des services web invoqués par l'application temps réel en prenant en compte les contraintes temps réel et les préférences imposées par les clients.

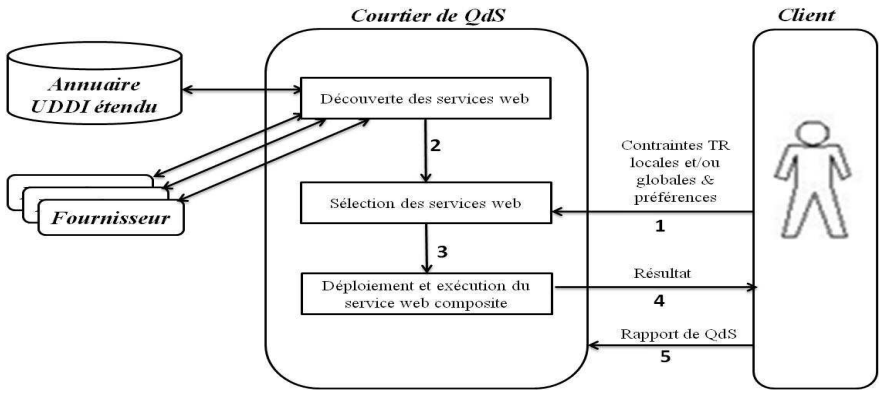

Figure 1 : Architecture des services web adaptée aux applications temps réel

Toutes les interactions avec le courtier de QdS peuvent être résumées dans les étapes suivantes :

- ***Etape 1 :*** Le client décrit les contraintes temps réel locales et/ou globales dans une requête de QdS. Cette requête sera par la suite prise en compte par le courtier de QdS.

- ***Etape 2 :*** Le courtier se charge en premier lieu de la découverte de services web en interrogeant l'UDDI étendu pour collecter des informations sur les services web disponibles (leurs descriptions de QdS, leurs descriptions WSDL, etc.) qui peuvent exécuter les tâches de l'application temps réel. Le résultat de la découverte consiste alors à définir pour chaque tâche un ensemble de services web aptes à l'exécuter.

- ***Etape 3 :*** En se basant sur les exigences décrites dans la requête du client, le courtier choisit un service web pour exécuter chaque tâche de l'application temps réel et ceci en appliquant une stratégie de sélection et en déroulant un algorithme de sélection. Le résultat de la sélection sera un ensemble de services web coopérés et inter-opérés formant ainsi un service web composite.

- ***Etape 4 :*** Le courtier se charge alors de déployer le service web composite et de l'exécuter dans un moteur d'exécution de services web. Le résultat de l'exécution du service web composite sera par la suite transféré au client.

- ***Etape 5 :*** En analysant le résultat reçu, le client peut envoyer un rapport de QdS contenant son avis sur le résultat de l'exécution du service composite. Ce rapport sera pris en compte par le courtier pour une prochaine sélection de services web.

3.2. Description de l'application

Nous disposons de l'application temps réel *A* « réservation de circuit touristique ». Elle est composée de l'ensemble de tâches $T_A = \{t_1, t_2, t_3, t_4\}$ avec :

- t_1 : représente la tâche réservation de vol,
- t_2 : représente la tâche réservation de voiture,
- t_3: représente la tâche réservation d'hôtel,
- t_4 : représente la tâche paiement en ligne.

Dès que la tâche de réservation de vol est effectuée, il s'agit d'exécuter en parallèle les deux tâches réservation de voiture et réservation d'hôtel. Une fois que ces deux tâches ont terminé l'exécution, il s'agit alors d'exécuter la tâche paiement en ligne.

Des contraintes de temps sont imposées localement sur les différentes tâches de l'application temps réel. Nous proposons ainsi que :

- La contrainte relative à la tâche « réservation de vol » est : la durée d'exécution de la tâche « réservation de vol » ne doit pas dépasser les 10mn ;
- La contrainte relative à la tâche « réservation de voiture » est : la durée d'exécution de la tâche « réservation de voiture » doit être inférieure à 8mn ;
- La contrainte relative à la tâche « réservation d'hôtel » est : la durée d'exécution de la tâche « réservation d'hôtel » ne doit pas dépasser les 5mn ;
- La contrainte relative à la tâche « paiement en ligne » est : la durée d'exécution de la tâche « paiement en ligne» doit être inférieure à 5mn.

Notons que les contraintes imposées localement peuvent être indépendantes du temps comme par exemple exiger que le budget pour la réservation de voiture ne doit pas dépasser les 200 euros, mais dans notre exemple, nous tenons compte uniquement des contraintes temporelles.

Cette application est décrite en détail dans la figure 2.

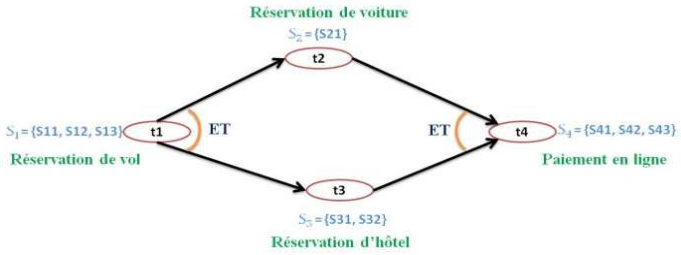

Figure 2 : Tâches de l'application « réservation de circuit touristique » faisant appel à des services web

Nous remarquons que pour une seule tâche, on peut trouver plusieurs services web aptes à l'exécuter tels que par exemple {S11, S12 ou S13} pour exécuter la tâche réservation de vol, de même pour les tâches réservation de voiture, réservation d'hôtel et paiement en ligne. Donc un besoin de choix est considéré pour déterminer quel service exécute quelle tâche. Pour ce faire, nous proposons de suivre la démarche suivante : il faut tout d'abord choisir la stratégie de sélection de services et ceci en fonction de la nature des contraintes de QdS (relatives à chaque tâche ou à l'ensemble des tâches), puis modéliser le problème de sélection en fonction de la stratégie choisie (selon la technique SAW ou bien comme un problème d'optimisation combinatoire) et enfin dérouler un algorithme de sélection de services web adéquat à la stratégie de sélection choisie.

3.3. Déroulement d'un algorithme de sélection

Nous proposons dans le cadre de cet article d'appliquer la stratégie de sélection locale. Le problème de sélection sera donc modélisé selon la technique SAW. En fait, cette technique se base généralement sur l'application des algorithmes inspirés de l'heuristique gloutonne qui a comme principe de faire, étape par étape, un choix optimum local, dans l'espoir d'obtenir un résultat optimum global.

En appliquant ce principe, nous proposons de dérouler l'algorithme suivant:

Début

Pour chaque ensemble de services S_i

Pour chaque service web S_{ij} appartenant à l'ensemble S_i

Calculer son score en appliquant la fonction de score relative à la technique SAW

Fin

Sélectionner le service qui a le score le plus élevé et qui respecte les contraintes temps réel et les préférences locales imposées par le client afin de lui affecter la tâche *i* qui appartient à l'ensemble T_A

Fin

Fin

3.4. Modélisation du problème de sélection selon la technique SAW

Dans ce qui suit, nous proposons de modéliser le problème de sélection selon la technique SAW. Cette technique est composée de deux phases : phase d'ajustement et phase d'affectation de poids.

- *Phase d'ajustement :* chaque service web candidat dans la sélection a son propre vecteur de QdS contenant les valeurs des critères de QdS optimisant le temps. Pour rendre les valeurs de chaque vecteur homogènes, nous appliquons les deux formules de normalisation (1) et (2) :

$$n_{ij}= \begin{cases} \frac{max_i(S_{ij})-S_{ij}}{max_i(S_{ij})-min_i(S_{ij})} & \textbf{(1)} \text{ Pour les critères à dimension décroissante tels que le temps de réponse et la latence} \\ \frac{S_{ij}-min_i(S_{ij})}{max_i(S_{ij})-min_i(S_{ij})} & \textbf{(2)} \text{ Pour les critères à dimension croissante tels que le débit, la capacité et la disponibilité} \end{cases}$$

Avec S_{ij} signifie que le service web *j* est assigné pour exécuter la tâche *i* (j=1..p et i=1..q)
Prenons par exemple le choix du service S_{11} ou S_{12} ou S_{13} pour exécuter la tâche t_1 réservation de vol. Voici les valeurs des critères de QdS qui caractérisent chaque service web :

$$V_{11} = \begin{pmatrix} TR_{11} = 10mn \\ LAT_{11} = 1mn \\ DEB_{11} = \frac{3}{30mn} \\ CAP_{11} = 2 \\ DIS_{11} = 60\,\% \end{pmatrix}$$

$$V_{12} = \begin{pmatrix} TR_{12} = 9mn \\ LAT_{12} = 3mn \\ DEB_{12} = \dfrac{5}{30mn} \\ CAP_{12} = 2 \\ DIS_{12} = 80\,\% \end{pmatrix} \qquad V_{13} = \begin{pmatrix} TR_{13} = 8mn \\ LAT_{13} = 1mn \\ DEB_{13} = 4/30mn \\ CAP_{13} = 3 \\ DIS_{13} = 70\,\% \end{pmatrix}$$

En appliquant les deux formules de normalisation (1) et (2), nous obtenons les trois vecteurs de valeurs normalisés suivants :

$$V'_{11} = \begin{pmatrix} TR'_{11} = 0 \\ LAT'_{11} = 1 \\ DEB'_{11} = 0 \\ CAP'_{11} = 0 \\ DIS'_{11} = 0 \end{pmatrix} \qquad V'_{12} = \begin{pmatrix} TR'_{12} = \dfrac{1}{2} \\ LAT'_{12} = 0 \\ DEB'_{12} = 1 \\ CAP'_{12} = 0 \\ DIS'_{12} = 1 \end{pmatrix}$$

$$V'_{13} = \begin{pmatrix} TR'_{13} = 1 \\ LAT'_{13} = 1 \\ DEB'_{13} = \dfrac{1}{2} \\ CAP'_{13} = 1 \\ DIS'_{13} = \dfrac{1}{2} \end{pmatrix}$$

Pour choisir quel est le service web qui sera assigné à la tâche t_1, nous passons à la deuxième phase de la technique SAW qui consiste à calculer le score de chaque service web et celui qui a le score le plus élevé et qui respecte les contraintes temps réel et les préférences imposées par le client est celui qui est assigné à la tâche t_1.

- *Phase d'affectation de poids :* Dans cette phase, le client peut exprimer ses préférences envers la QdS des services web à sélectionner en affectant un poids à chaque critère de QdS. Puis, le courtier de QdS calcule le score de chaque service web qui représente une somme pondérée des valeurs des critères de QdS normalisées. A ce niveau, le courtier procède au choix du service adéquat à exécuter la tâche t_1 en sélectionnant le service qui a le score le plus élevé et qui respecte aussi les contraintes temps réel qui sont imposées localement par le client pour chaque tâche (pour notre exemple le client propose que la durée d'exécution de la tâche t_1 ne doive pas dépasser les 10mn). S'il y a plusieurs services ayant la même valeur du score maximal, alors on choit le service qui respecte mieux les contraintes temps réel qui seront appliquées à la tâche à laquelle il est affecté. Si pour une tâche donnée, aucun service ne respecte les contraintes temps réel imposées par le client, une exception d'exécution sera

levée et le courtier proposera au client de modifier ses contraintes temps réel ou bien la valeur des poids affectée à chaque critère.

Voici dans ce qui suit la valeur des poids relatifs aux critères temps de réponse, latence, débit, capacité et disponibilité.

$W(TR) = 0{,}7$; $W(LAT) = 0{,}1$; $W(DEB) = 0$; $W(CAP) = 0$; $W(DISP) = 0{,}2$

Le score de chaque service web est calculé en appliquant la formule suivante :

$$\text{Score } (S_{ij}) = \sum_{x=1}^{5}(w_{x} * v_{x}) \quad (2)$$

Avec :

- x représente un indice d'un critère de QdS et $1 \leq x \leq 5$ puisque nous considérons que la sélection des services web sera basée sur les cinq critères de QdS optimisant le temps que nous avons définis dans la section 3.2 (pour le temps de réponse, x=1 ; pour la latence, x=2 ; pour le débit x=3 ; pour la capacité, x=4 et pour la disponibilité, x=5).
- W_x est un poids défini par le client pour définir sa préférence envers le critère d'indice x. La somme des poids doit être toujours est égale à 1.
- v_x est la valeur normalisée du critère d'indice x spécifiant le service S_{ij}. Cette valeur est extraite du vecteur V'_{ij}.

Pour notre exemple, nous calculons le score de S_{11} comme suit :

$\text{Score } (S_{11}) = \sum_{x=1}^{5}(w_{x} * v_{x})$

$\text{Score } (S_{11}) = w_{TR} * v_{TR} + w_{LAT} * v_{LAT} + w_{DEB} * v_{DEB} + w_{CAP} * v_{CAP} + w_{DISP} * v_{DISP}$

$\text{Score } (S_{11}) = 0.7 * 0 + 0{,}1 * 1 + 0 * 0 + 0 * 0 + 0.2 * 0$

$\text{Score } (S_{11}) = 0{,}1$

De la même manière, nous calculons le score de S_{12} et le score de S_{13} :

$\text{Score } (S_{12}) = 0.7 * \frac{1}{2} + 0{,}1 * 0 + 0 * 1 + 0 * 0 + 0.2 * 1$

$\text{Score } (S_{12}) = 0{,}55$

$\text{Score } (S_{13}) = 0.7 * 1 + 0{,}1 * 1 + 0 * \frac{1}{2} + 0 * 1 + 0.2 * \frac{1}{2}$

$\text{Score } (S_{13}) = 0{,}9$

Nous remarquons que le score de S_{13} est le plus grand, en plus la contrainte temps réel exprimée par le client est respectée. Ceci s'explique par le fait que le temps de réponse du service S_{13} est égal à 8mn et il est inférieur à 10mn qui représente la durée maximale d'exécution de la tâche t_1 qu'impose le client. Donc le service S_{13} est celui qui est assigné à la tâche t_1 afin de l'exécuter.

On applique le même principe de sélection pour choisir le service web qui exécute les tâches t_2, t_3 et t_4. On aura à la fin un seul service web invoqué pour exécuter chaque tâche.
La figureure 3 décrit l'ensemble des services web invoqués par l'application temps réel accompagnés par les vecteurs contenant les valeurs des critères de QdS optimisant le temps.

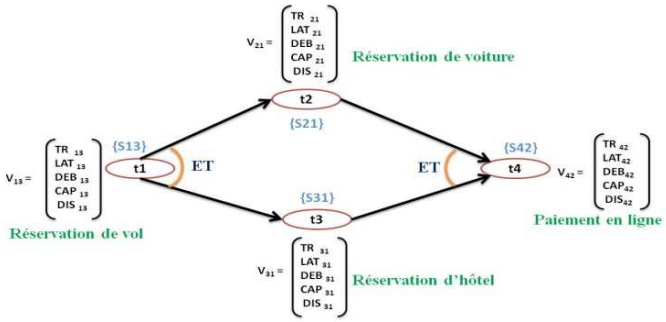

Figure 3 : Services web appelés par l'application « réservation de circuit touristique » suivis de leurs valeurs de critères de QdS optimisant le temps

Par ailleurs, les critères de qualité qui décrit chaque service peuvent être aussi appliqués pour évaluer la QdS de l'ensemble des services et qui forment un service web composite.

3.5. Critères de qualité pour les services composites

Lorsqu'on a affaire à évaluer la QdS d'un service composite et voir s'il répond à des contraintes relatives à l'ensemble des services composants (comme par exemple vouloir un service composite dont la disponibilité est supérieure à 80 % et la durée d'exécution de bout en bout ne dépasse pas 25mn), on a intérêt à calculer la valeur des critères de QdS relative au service composite. Le calcul de la valeur de ces critères s'appuie sur l'application des fonctions d'agrégation [4, 12 et 13]. La formulation de ces fonctions (tantôt une somme, tantôt une moyenne, tantôt un produit, etc.) dépend de la combinaison appliquée aux services web lors de la composition. Les services web peuvent être combinés de différentes manières parmi lesquelles nous distinguons la combinaison séquentielle, parallèle, à condition et la combinaison utilisant une boucle [4].

Et comme il n'existe pas de fonction standard d'agrégation à appliquer nous ne pouvons pas donner de formule générique pour calculer la valeur des critères pour le service web composite. Dans notre étude, nous avons proposé de vérifier si les contraintes relatives à la disponibilité et au temps de réponse sont respectées par la

totalité des services composants et nous avons donc proposé une formulation des fonctions d'agrégation pour calculer le temps de réponse et la disponibilité relatifs à un service composite.

Pour définir les fonctions d'agrégation, nous avons utilisé les paramètres suivants :

C : désigne le service web composite qui est représenté sous forme de graphe orienté.

S_i : est un service web composant qui appartient au service web composite *C* et qui exécute la tâche t_i de l'application temps réel *A*.

Ces fonctions seront présentées dans le tableau 1.

	Temps de réponse	Disponibilité
Séquentiel	$\mathrm{TR(C)} = \sum \mathrm{TR(S_i)}$	$\mathrm{DISP(C)} = \prod_{i=1}^{N} (e^{\mathrm{DISP(S_i)}})$
Parallèle	$\mathrm{TR(C)} = \mathrm{TR(S_i)} + \mathrm{Max}(\mathrm{TR(S_{i+1})}, \dots, \mathrm{TR(S_n)}) + \mathrm{TR(S_{n+1})}$	$\mathrm{DISP(C)} = e^{\mathrm{DISP(S_i)}} * e^{\mathrm{Min}(\mathrm{DISP(S_{i+1})},\dots,\mathrm{DISP(S_n)})} * e^{\mathrm{DISP(S_{n+1})}}$
Conditionnel	$\mathrm{TR(C)} = \sum \mathrm{TR(S_i)}$	$\mathrm{DISP(C)} = \prod_{i=1}^{N} (e^{\mathrm{DISP(S_i)}})$
Répétitif	$\mathrm{TR(C)} = n * \mathrm{TR(S_i)}$	$\mathrm{DISP(C)} = (e^{\mathrm{DISP(S_i)}})^n$

Tableau 1 : Fonctions d'agrégation pour le calcul du temps de réponse et de la disponibilité

Pour le calcul du temps de réponse pour le modèle séquentiel, il faut que chaque service web attende la fin de l'exécution des services qui le précèdent. Pour le modèle parallèle, il est nécessaire d'attendre tous les résultats exécutés en parallèle, d'où la présence d'une fonction Max. Pour le modèle conditionnel, nous calculons le temps de réponse selon la condition imposée. Et pour le modèle répétitif, nous prenons en considération combien de fois un service web peut être exécuté.
Par ailleurs, pour le calcul de la disponibilité pour le modèle séquentiel, nous effectuons un produit de toutes les valeurs puisque la valeur de la disponibilité de chaque service composant est un pourcentage. Pour le modèle parallèle, nous choisissons d'appliquer la fonction Min pour prendre en considération la disponibilité des services web exécutés en parallèle. Pour le modèle conditionnel,

nous calculons la disponibilité selon la condition. Et pour le modèle répétitif, nous prenons en considération combien de fois un service web peut être exécuté.

Conclusion

Dans ce papier nous avons présenté le processus de sélection des services web lorsqu'on a affaire à sélectionner parmi plusieurs services ceux qui exécutent une application temps réel et qui respectent les contraintes de temps imposées. Ainsi nous avons présenté les différentes stratégies de sélection de services web en fonction de la nature des contraintes de QdS imposées (relatives à chaque tâche ou bien à l'ensemble des tâches). Ensuite pour chaque stratégie, nous avons défini comment modéliser le problème de sélection de services web basée sur la QdS

D'après notre étude, il ressort que la prise en compte des contraintes de temps au niveau des services se fait à travers les critères de QdS, de ce fait nous avons distingué les différentes classifications des critères de QdS afin de recenser les critères de QdS qui peuvent influencer le respect des contraintes de temps. Afin de mettre en œuvre une application temps réel exécutée à l'aide des services web, nous avons présenté une étude de cas qui consiste à présenter l'architecture des services web adéquate aux applications temps réel, à présenter l'exemple de l'application « réservation de circuit touristique » qui est exécutée par des services web, à modéliser le problème de sélection des services qui respectent les contraintes temporelles et enfin à proposer une formulation des fonctions d'agrégation pour calculer le temps de réponse et la disponibilité pour un service web composite.

Pour les travaux futurs, nous prévoyons d'étudier les contraintes de QdS globales et de modéliser ainsi le problème de sélection comme un problème d'optimisation combinatoire, de définir une fonction coût relative aux applications temps réel et d'étudier la combinaison des contraintes locales et globales.

Références

[1] Benatallah, B., Dumas, M., Sheng, Q.Z. and Ngu, A.. Declarative Composition and Peer-to-Peer Provisioning of dynamic Web Services. In proceedings of IEEE ICDE'02, San Jose, USA : February, 297- 308. (2002)

[2] ISO. Qualité, Normes et ISO 9000. http://www.jalix.org/ressources/miscellaneous/~vrac/SI/ISO/#_Toc411087330. (2005)

[3] Jaeger, M.C. and Muhl, G.. Soft real-time aspects for service-oriented architectures. In proceedings of the 8th IEEE International Conference on E-

Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, San Francisco, CA, 5-5. (2006)

[4] Liu, Y., Ngu, A. and Zeng, L.. QoS Computation and Policing in Dynamic Web Service selection. In proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, New York, USA ; May, 66, 73. (2004)

[5] Pisinger, D.. A minimal algorithm for the Multiple-Choice Knapsack Problem. In proceedings of the European Journal of Operational Research, 2-8 June, 394-410. (1995)

[6] Ran, S.. A Model for Web Services Discovery with QoS. ACM SIGecom Exchange. Volume 4, issue 1, 1-10. (2003)

[7] Wikipédia. Algorithme glouton. http://fr.wikipedia.org/wiki/Algorithme_glouton. (2008).

[8] Wang, X., Vitvar,T., Kerrigan, M. and Toma, I.. A QoS-aware Selection Model for Semantic Web Services. In Proceedings of the 4th International Conference on Service Oriented Computing ICSOC'06, Chicago, USA: 4-7 December, 390-401. (2006)

[9] Yu, T. and Lin, K.J.. A Broker-Based Framework for QoS-Aware Web Service Composition. In Proceedings of 2005 IEEE International Conference on E-Technology, E-Commerce and E-Service (EEE05), Hong Kong, China: 29 March-1 April, 22-29. (2005)

[10] Yu, T. and Lin, K.J.. Service Selection Algorithms for Composing Complex Services with multiple QoS Constraints. In Proceedings of the 3rd International Conference on Service Oriented Computing (ICSOC2005), Amsterdam, The Netherlands : 12-15 December, 130-143. (2005)

[11] Yu, T. and Lin, K.J.. Service Selection Algorithms for Web Services with End-to-end QoS Constraints. In Proceedings of the 2004 IEEE Conference on Electronic Commerce (CEC04), San Diego, California: 6-9 July, 129-136. (2004)

[12] Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J. and Sheng, Q.Z.. Quality Driven Web Services Composition. In proceedings of the 12th International World Wide Web Conference (WWW2003), Budapest, Hungary: 20-24 May, 411-421. (2003)

[13] Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J. and Chang, H.. QoS-Aware Middleware for Web Services Composition. IEEE transactions on software engineering. Volume 30, issue 5, 311-327. (2004)