Méthode de sélection intentionnelle des services, en fonction des exigences non-fonctionnelles des agents métier

Assia Ait Ali Slimane

Université Paris I – Panthéon-Sorbonne

Méthode de sélection intentionnelle des services, en fonction des exigences non-fonctionnelles des agents métier

Thèse de doctorat soumise pour l'obtention du titre de :

Docteur de l'université Paris I – Panthéon-Sorbonne

Spécialité : Informatique

Présentée par :

Assia AIT ALI SLIMANE

Soutenue le 20/12/2012 devant le jury composé de :

Mr David NACCACHE Président de jury
Mme Corine CAUVET Rapporteur
Mme Camille ROSENTHAL-SABROUX Rapporteur
Mme Colette ROLLAND Membre de jury
Mme Carine SOUVEYET Directeur de thèse

À la mémoire de mon frère

REMERCIEMENTS

Je tiens tout d'abord à remercier sincèrement les membres du jury :

M^r David Naccache, Professeur à l'université Paris II Panthéon-Assas, d'avoir accepté de présider le jury de ma soutenance.

M^me Corine Cauvet, Professeur à l'université Aix-Marseille 3, et M^me Camille Rosenthal-Sabroux, Professeur à l'université Paris-Dauphine, d'avoir accepté d'être les rapporteurs de mon mémoire.

M^me Colette Rolland, Professeur à l'Université de Paris 1 Panthéon – Sorbonne, de m'avoir accueillie dans son équipe, pour ses conseils, et d'avoir accepté de faire partie de mon jury.

M^me Carine Souveyet ma directrice de thèse, Professeur à l'université Paris1 Panthéon-Sorbonne, d'avoir accepté la direction scientifique de ma thèse, ses conseils, son aide et sa ténacité tout au long de cette thèse.

Je remercie également M^me Manuele Kirsch Pinheiro, Maitre de Conférence à l'Université de Paris 1 Panthéon – Sorbonne, pour son aide précieuse, ses encouragements continus et sa disponibilité.

Mes remerciements vont également à M^r Camille Salinesi, Professeur - Directeur du Centre de Recherche en Informatique à l'Université de Paris 1 Panthéon – Sorbonne, pour ses conseils.

Je remercie amicalement tous les membres de l'équipe du Centre de Recherche en Informatique pour leur soutien et leur encouragement. En particulier, Adrian pour le développement du prototype.

Je remercie chaleureusement mes amis pour leurs amitiés et leurs encouragements.

Enfin, je tiens à remercier ma famille et ma belle- famille pour leur soutien et leur patience, et particulièrement Habbib pour sa disponibilité, sa patience et son soutien continu, sans lui cette fin de thèse aurait été beaucoup plus dure.

« Détruire peut être l'œuvre d'une seule journée. Construire peut être le fruit d'un travail long et acharné »

Winston Churchill

Table des matières

Table des matières

CHAPITRE 1 : INTRODUCTION

1.1 *CONTEXTE*

La thèse s'inscrit dans le domaine de l'ingénierie des services.

L'informatique orientée service (SOC) est avant tout une façon de concevoir des systèmes logiciels orienté-fonction (*function-driven*) (Rolland et *al.*,2008). Les services exécutent des fonctions implémentées par le logiciel et encapsulées dans des interfaces formelles, lesquelles permettent de communiquer avec d'autres interfaces afin d'exécuter une fonction de plus haut niveau. L'interface d'un service est très importante, car c'est la seule partie du service qui est exposée au client.

Le client peut être aussi bien une application, un autre service, qu'un utilisateur humain. Ce dernier cas s'est nettement développé grâce à la prolifération du commerce électronique, à travers les applications *Business-to-client* (B2C) (*e.g.* suivi du trafic) et les *Business-to-Business* (B2B) (*e.g.* facturation) (Casati&Shan,2001) (Medjahed et *al.*,2003). On appelle ces services, dont les clients sont des utilisateurs humains plutôt que des applications, les « services dédiés à l'utilisateur ». Le développement des services dédiés à l'utilisateur témoigne de l'expansion de la notion de service au-delà de l'univers purement technique, traditionnellement soutenu par la communauté de recherche SOC. La notion de service est utilisée également dans le monde de la gestion (Piccinelli et *al.*,2003), dans lequel le service est considéré comme un service métier, correspondant à un groupement logique de composants requis pour satisfaire une demande des agents métier. Aussi, l'utilisateur final peut avoir des difficultés, d'une part, à exprimer ses besoins, et d'autre part à exploiter ces services, car il doit faire face à des notions techniques qu'il ne maîtrise pas forcément. Pour cela, plusieurs approches de l'ingénierie des exigences (*RE-Requirements Engineering*) (Penserini&Mylopoulos,2005) (Stollberg&Norton,2007) (Da Silva Santos et al.,2009) (Rolland et *al.*,2008) proposent d'utiliser la notion de but pour exprimer ce que l'utilisateur souhaite voir réaliser par un service. L'utilisation du but vise à hisser la définition des services à un haut niveau d'abstraction, facilitant ainsi son utilisation et sa compréhension par l'utilisateur final (Da Silva Santos et al.,2009) (Rolland et *al.*,2008) (Rolland et *al.*,2010). De plus, ces auteurs proposent de modéliser des services à forte variabilité, en définissant des

compositions de service orientées but. Celles-ci permettent de répondre à l'utilisateur final, de différentes manières.

Généralement, un service est défini en termes de son comportement fonctionnel, c'est-à-dire de ce que le service fait ou réalise comme fonctionnalité. Cependant, cette définition ne décrit pas entièrement le comportement du service. Ainsi, O'Sullivan et ses collègues (O'Sullivan et *al.*,2002) considèrent que la description d'un service n'est complète que lorsque ses qualités sont exprimées. Par le terme de qualité de service (QoS-Quality of Service) sont référées les propriétés non-fonctionnelles d'un service logiciel (Aiello&Giorgini,2004). Les propriétés non-fonctionnelles, telles que le temps d'exécution, sont des éléments importants pour déterminer la facilité d'utilisation et l'utilité d'un service (Aiello&Giorgini,2004).

Le terme de qualité de service (*QoS- Quality of Service*) est originaire de la communauté des réseaux et de celle des télécommunications (Clark et *al.*,1992) (ITU,1994) (Cruz,1995) (FIPA,2002) (Tian,2003), lesquelles définissent la qualité de service comme les paramètres liés aux caractéristiques du réseau, tels que la bande passante et la latence. Communément, selon le W3C (W3C,2003), la qualité de service se réfère aux propriétés non-fonctionnelles assurées par des services Web, telles que la performance (qui englobe le débit, le temps de réponse et le temps d'exécution), la fiabilité, ainsi que les exigences reliées au réseau. Ainsi, la qualité de service correspond souvent à des métriques calculées sur les services logiciels (Zeng et *al.*,2003) (Maximilien&Singh,2004) (Dobson et *al.*,2005)(Mohabbati et *al.*,2011).

Néanmoins, la description de la qualité de service, définie dans l'univers SOC, est de bas niveau (*e.g.*un OWL-QoS (Zhang et *al.*,2009)). Elle est difficilement compréhensible par les utilisateurs finaux qui doivent utiliser cette qualité pour choisir parmi plusieurs services. Sachant que la notion de qualité devient également familière dans le monde de la gestion (Batagan et *al.*,2009), celle-ci est adaptée au niveau métier, en décrivant la qualité associée aux services métier. La qualité des services dédiés à l'utilisateur est également orientée vers des utilisateurs humains, plutôt qu'à des applications. Plusieurs approches de l'ingénierie des exigences (Aiello&Giorgini,2004) (Penserini et *al.*,2006) (Rohleder,2009) (AitAliSlimane et *al.*,2009) (AitAliSlimane et *al.*,2011) proposent de définir la qualité de service comme l'ensemble des buts qualité (dit *soft goal*) que les services peuvent partiellement satisfaire.

Les deux communautés SOC et RE sont unanimes sur le fait que la qualité de service joue un rôle prépondérant dans la sélection des services (Aiello&Giorgini,2004) (O'Sullivan et *al.*,2002). La sélection de service est l'étape permettant de différencier les services fournissant la même fonctionnalité, en se basant sur la qualité de service (Maximilien&Singh,2004b).

Les approches SOC proposent généralement un processus de sélection composé de deux parties principales : (1) une modélisation de la qualité de service, laquelle est généralement effectuée en proposant des ontologies de qualité de service (Maximilien&Singh,2004b) (Dobson et *al.*,2005) (Giallonardo&Zimeo,2007) (Kritikos&Plexousakis,2007). Celle-ci permet d'effectuer une correspondance sémantique entre les QoS publiées par les fournisseurs et les préférences QoS des clients. La modélisation de la qualité de service se fait également en utilisant la logique floue (Huang et *al.*,2007)(Li et *al.*, 2011)(Sun et *al.*,2011) ; et (2) un algorithme de sélection, lequel permet dynamiquement, à l'exécution, de sélectionner les services implémentant une fonctionnalité donnée, en fonction de leurs QoS. Ces approches se partagent, notamment, entre celles qui utilisent des méthodes multicritères d'aide à la prise de décision (*MCDM -Multiple Criteria Decision Making*) (Zeng et *al.*,2003) (Wang et *al.*,2006) (Herssens et *al.*,2008), de la programmation linéaire (Zeng et *al.*,2003), ou encore celles qui utilisent des heuristiques (Jaeger et *al.*,2005) (Yu et *al.*,2007) (Comes et *al.*,2010). D'autres approches exploitent les expressions floues de la qualité de service pour effectuer la sélection de services (Chen et *al.*, 2003) (Li et *al.*, 2011).

Les approches de l'ingénierie des exigences, quant à elles, sélectionnent les services statiquement, lors de la conception du système. Elles permettent de choisir les services dont les qualités correspondent aux exigences non-fonctionnelles des utilisateurs finaux (Penserini et *al.*,2006)(Ma et *al.*,2009). Ma et ses collègues (Ma et *al.*,2009), par exemple, proposent une ontologie de QoS des services métier en se basant sur le cadre d'application NFR (Chung et *al.*,2000) et utilisent une méthode multicritères d'aide à la prise de décision pour sélectionner les services.

Nous remarquons que la sélection des services dans la communauté SOC est effectuée à l'exécution des services logiciels visant notamment leur usage lors d'une composition de services. Toutefois, ni le processus de sélection en-soi, ni l'interface des services logiciels sélectionnés, ne restent compréhensibles aux utilisateurs finaux. Les approches de l'ingénierie des exigences, quant à elles, font la sélection des services de haut niveau durant la conception de système. Néanmoins, elles ne considèrent pas la perspective logicielle de ces services.

L'étape de sélection permet de choisir des services, notamment des services de haut niveau, dont les descriptions sont basées sur les buts, au mieux des QoS (Rolland et *al.*,2008). En outre, ces services de haut niveau peuvent encapsuler différentes compositions de services permettant de réagir différemment selon le contexte métier. Ces services permettent donc d'introduire de la variabilité dans la manière de réaliser le but du service demandé. Cette variabilité consiste à identifier les différentes variantes capables de réaliser le but de ce

service. Chacune de ces variantes peut contribuer différemment à la qualité globale du service. L'utilisateur final a besoin d'être guidé, non seulement dans la sélection et l'exécution des services de haut niveau, mais également dans le choix de leurs variantes les plus appropriées. On appelle la configuration de service le choix des variantes appropriées aux QoS souhaitées par l'utilisateur.

Les communautés SOC et RE considèrent que plusieurs configurations, répondant de différentes manières aux mêmes exigences fonctionnelles, peuvent fournir diverses QoS (Xiong et *al.*,2008) (Rohleder,2009) (Acher et *al.*,2010).

D'un point de vue SOC, Teije et ses collègues (Teije et *al.*,2004) proposent de configurer le service Web composite, alors que (Xiong et *al.*,2008) permettent de choisir et d'invoquer dynamiquement des services pour des *Workflows*. Le choix des services Web se base aussi bien sur leurs descriptions, que sur les contraintes du processus (Verma et *al.*,2005). Acher et ses collègues (Acher et *al.*,2010) parlent de configuration de *Workflows*, en assemblant des services paramétrés dans un *Workflow*. Ils proposent d'organiser ces services comme des éléments d'une ligne de produit, et définissent une famille de services comme un ensemble de préoccupations qui exhibent de la variabilité, lesquelles peuvent être représentés par les langages de modélisation de la variabilité. Aussi, d'autres auteurs (Mohabbati et *al.*,2011) proposent une approche de configuration de ligne de produit logiciel orienté service (SOSP-Service-Oriented Software Product Lines), en optant pour une méthode d'optimisation linéaire pour la sélection du service optimal.

D'un point de vue RE, le processus de configuration concerne principalement la configuration de ligne de produit logiciel (*SPL-Software Product Line*). Celle-ci consiste mettre en avant les éléments variants (aussi appelées « points de variation ») de la ligne de produit. Ensuite, il s'agit de construire des produits individuels (on parle aussi de « processus de configuration ») en figeant certains choix vis-à-vis des éléments variables, tout en tenant compte des restrictions concernant la compatibilité de ces éléments. Dans le cas de plusieurs configurations conformes aux exigences et aux restrictions, il faut alors effectuer un choix optimal selon les critères de qualité. Les approches RE s'intéressent d'une part à la configuration des processus métier de haut niveau (Lapouchnian, et *al.*,2007), et d'autre part la configuration des lignes de produit (Niemelä&Immonen,2007) (Rohleder,2009). Particulièrement, les approches de ligne de produit (Deelstra et *al.*,2004) (White et *al.*,2009) (Djebbi,2011) proposent de dériver des configurations valides, qui doivent considérer les dépendances fonctionnelles pouvant exister entre les éléments de la ligne de produits.

Ainsi, deux visions s'opposent : d'une part, une vision technique, tournée vers l'exécution des services logiciels, mais qui reste peu compréhensible aux utilisateurs finaux. D'autre part, une vision de plus haut niveau qui assume le point de vue de ces utilisateurs finaux et de leurs exigences, mais qui reste déconnectée des services logiciels et de leur exécution. Cette même séparation est également perceptible au niveau de la qualité de service et de sa description. D'un côté, nous avons une description de la qualité de service qui est très technique et directement liée aux métriques utilisées pour mesurer ces qualités, lors de l'exécution des services logiciel. D'un autre côté, nous avons des utilisateurs finaux qui ont des exigences non-fonctionnelles, lesquelles ne sont pas forcement exprimées dans ces mêmes termes quantitatifs que les métriques utilisées par les fournisseurs des services logiciels.

1.2 *PROBLEMATIQUE*

Face au contexte général présenté ci-dessus, quatre questions principales se posent, à savoir :

1) Comment rendre compréhensibles aux utilisateurs finaux la description des services logiciels et celle de leurs qualités ?

2) Comment permettre aux fournisseurs et aux utilisateurs finaux d'un même domaine de partager leur compréhension concernant la qualité des services de haut niveau et celle des services logiciels ?

3) Comment aider les utilisateurs finaux à sélectionner et à exécuter des services de haut niveau, qui correspondent au mieux à leurs exigences non-fonctionnelles ?

4) Comment assurer une configuration des services de haut niveau, guidée par les exigences non-fonctionnelles, qui soit exécutable ?

Ces quatre questions soulignent quatre problèmes majeurs : (1) la discordance conceptuelle entre les exigences (fonctionnelles et non-fonctionnelles) des utilisateurs finaux et les services logiciels et leurs qualités ; (2) le consensus qualité entre les fournisseurs et les utilisateurs finaux ; (3) la sélection et l'exécution des services de haut niveau, en fonction des exigences non-fonctionnelles des utilisateurs finaux ; et (4) la configuration des services de haut niveau, en fonction des exigences non-fonctionnelles des utilisateurs finaux.

1.2.1 LA DISCORDANCE CONCEPTUELLE

La discordance conceptuelle concerne : (1) la discordance conceptuelle entre les exigences fonctionnelles des utilisateurs finaux et les services logiciels ; et (2) la discordance conceptuelle entre les exigences non-fonctionnelles et la qualité des services logiciels.

1.2.1.1 *LA DISCORDANCE CONCEPTUELLE ENTRE LES EXIGENCES FONCTIONNELLES ET LES SERVICES LOGICIELS*

Dans l'approche orientée service, les services exécutent des fonctions implémentées par le logiciel et encapsulées dans des interfaces formelles qui fournissent les signatures des opérations disponibles. Celles-ci permettent de communiquer avec d'autres interfaces (dans une composition) afin d'exécuter une fonction de plus haut niveau.

Néanmoins, l'expression des services par ces interfaces n'est pas forcément compréhensible par un utilisateur métier, voir un gestionnaire ou un décideur. Ces derniers ne sont pas intéressés par les détails techniques exposés par les interfaces, exprimées par exemple en WSDL. De plus, ces interfaces décrivent les fonctionnalités offertes par un service sans mentionner en quoi celles-ci peuvent répondre aux exigences fonctionnelles des utilisateurs finaux. Comme le montre la Figure 1, il existe alors un problème de mise en correspondance entre les exigences fonctionnelles des utilisateurs finaux et les services logiciels, déjà identifié dans (Kaabi,2007) (Rolland et *al.*,2008).

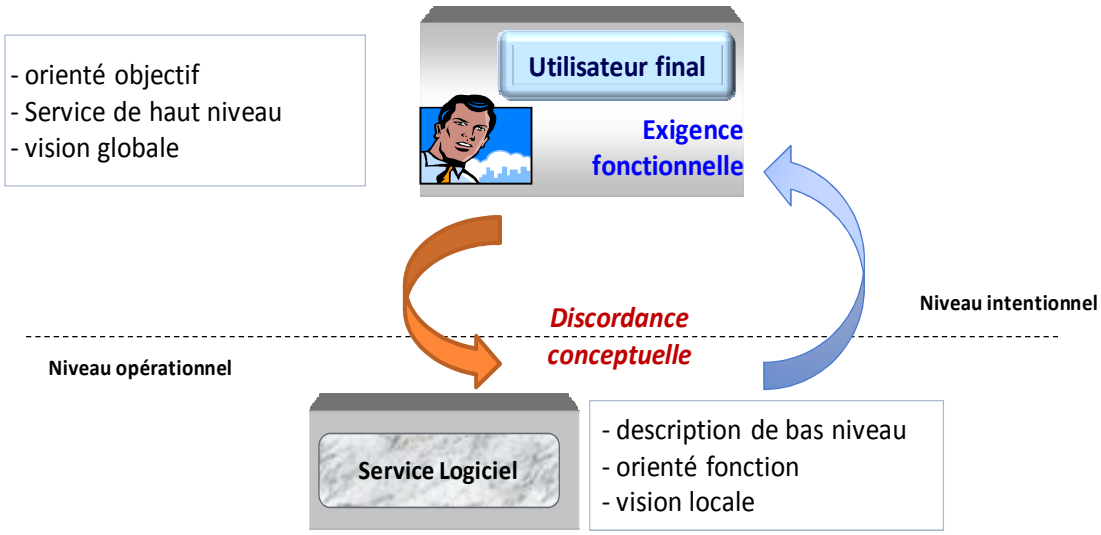

Figure 1. Problème de discordance conceptuelle entre les exigences fonctionnelles des utilisateurs finaux et le service logiciel (Kaabi,2007)

Cette difficulté de mise en correspondance est due à la discordance conceptuelle, laquelle tient principalement à la disparité des langages utilisés d'une part par les fournisseurs pour décrire leurs services, et d'autre part, par les utilisateurs finaux pour exprimer leurs besoins. En effet, d'un coté, le fournisseur se place à un niveau opérationnel, il se focalise sur le « quoi » et le « comment » des opérations. De l'autre coté, les utilisateurs finaux se placent au niveau intentionnel et possèdent un certain nombre d'objectifs qu'ils souhaitent satisfaire à travers l'utilisation de ces services.

Comme Stollberg, Rolland, Da Silva Santos et leurs collègues (Stollberg&Norton,2007) (Rolland&Kaabi,2007) (Da Silva Santos et al.,2009) (Rolland et *al.*,2010), nous pensons que la description du service doit être orientée vers l'utilisateur final. Ce problème de discordance conceptuelle (*cf.* Figure 1) est traité par Kaabi, Rolland et leurs collègues (Kaabi,2007) (Rolland et *al.*,2007) (Rolland et *al.*,2010). Ces auteurs proposent de combler le fossé existant entre les exigences fonctionnelles des agents métier et la description des services logiciels, par une définition de haut niveau des services de manière à ce que le monde du métier les comprennent, et par une correspondance entre ces services et les services logiciels qui les réalisent. Par contre cette mise en correspondance faite par Rolland et ses collègues (Kaabi et *al.*,2004) (Rolland&Kaabi,2007) (Rolland et *al.*,2007) (Rolland et *al.*,2010) se limitent aux aspects fonctionnels des services. Les aspects non-fonctionnels, et donc la qualité de service, ne sont pas traités par ces auteurs.

1.2.1.2 *LA DISCORDANCE CONCEPTUELLE ENTRE LES EXIGENCES NON-FONCTIONNELLES ET LA QUALITE DES SERVICES LOGICIELS*

Dans la communauté SOC, la qualité de service, à travers ses informations techniques, se focalise sur le 'Quoi' et le 'Comment'. Ces qualités sont orientées développeurs et concepteurs de services, lesquels étudient plusieurs aspects techniques correspondant aux diverses techniques d'implémentation de la qualité, de son calcul, et de son observation. L'expression de la qualité de service par des informations techniques et des métriques (*i.e.* « latence ») est souvent incompréhensible ou difficilement exploitable par les utilisateurs finaux (ou les clients du service). Une telle qualité ne mentionne pas la raison de son observation, ni en quoi permet-elle de satisfaire les exigences non-fonctionnelles des utilisateurs finaux. Ces derniers expriment leurs besoins en termes de buts à atteindre, mais non en termes de métriques ou de techniques nécessaires. Aussi, l'évaluation de la qualité de service effectuée par la communauté SOC exprime des valeurs techniques souvent quantitatives (*i.e.* « 250 millisecondes »), sans pour autant prendre en compte si ces valeurs vont satisfaire, peu satisfaire ou ne pas satisfaire les exigences non-fonctionnelles des utilisateurs finaux.

En d'autres termes, comme le montre la Figure 2, il subsiste un problème de mise en correspondance entre les exigences non-fonctionnelles des utilisateurs finaux (exprimées au niveau intentionnel) et la qualité des services logiciels (exprimée au niveau opérationnel). La difficulté de la mise en correspondance est due à la discordance conceptuelle, laquelle tient également à la disparité des modèles de représentation de la qualité dans lesquels les deux parties prenantes, à savoir les fournisseurs et les utilisateurs finaux, ont l'habitude de

s'exprimer : les premiers (les fournisseurs) se placent au niveau opérationnel et parlent de « latence » et d'évaluation quantitative de type « 250 millisecondes », tandis que les derniers (les utilisateurs finaux) se placent au niveau intentionnel et s'expriment en termes de « rapidité » et d'évaluation qualitative, telle que « très satisfait ».

Tout comme Aiello, Penserini, Ma et leurs collègues (Aiello et *al.*,2004) (Penserini et *al.*,2006) (Ma et *al.*,2009), nous pensons que l'expression de la qualité de service selon le point de vue de l'utilisateur final est nécessaire. Ce problème implique qu'il est important de rassembler des éléments conceptuellement différents et d'offrir un mécanisme d'abstraction permettant d'une part de naviguer dans les différents niveaux de granularité, et d'autre part de relier les mesures quantitatives calculées par la communauté SOC à l'évaluation qualitative assimilées par les utilisateurs finaux.

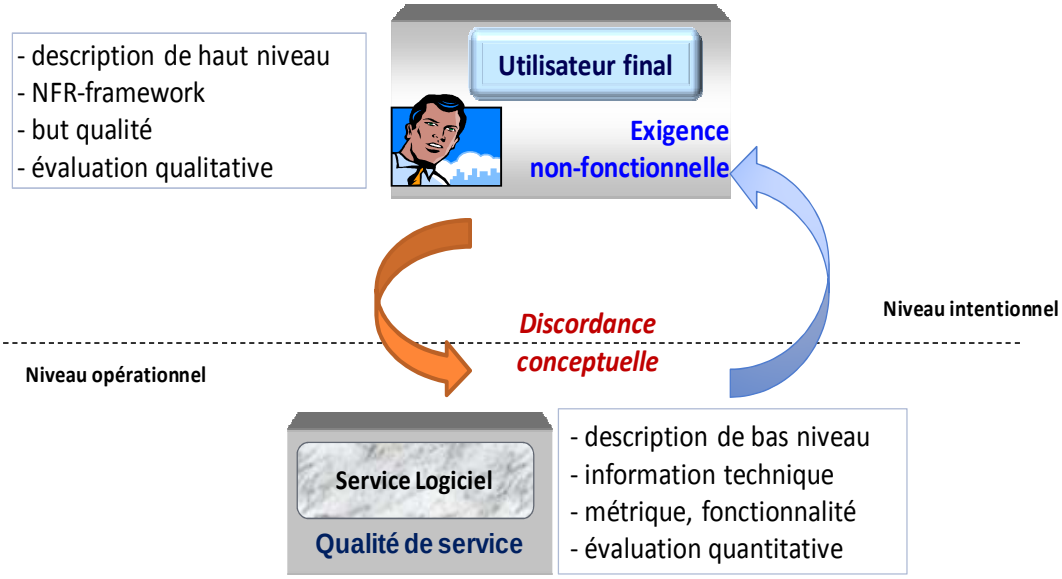

Figure 2. Problème de discordance conceptuelle entre les exigences non-fonctionnelles des utilisateurs finaux et la qualité du service logiciel

1.2.2 LE CONSENSUS QUALITE

Les fournisseurs, lors de la publication de la qualité de leurs services, ont tendance à surestimer leurs offres de qualité. Ils ne partagent pas forcément le même vocabulaire qualité et ils ne donnent pas nécessairement la même signification aux valeurs de satisfaction, car ce qui est « pas cher », « très fiable » ou «très sécurisé » pour l'un ne l'est pas forcément pour l'autre. Ran (Ran,2003), repris plus tard par Fedosseev (Fedosseev,2004), propose, par exemple, de rajouter le rôle de certificateur de qualité (*i.e. Web Service QoS Certifier*) au modèle SOA. Le certificateur de qualité permet de vérifier les offres de qualité des fournisseurs de service Web, afin d'améliorer la crédibilité de ces fournisseurs.

Lorsqu'on se place du point de vue de l'utilisateur final, il faut considérer également l'expression de ses exigences non-fonctionnelles, lesquelles doivent correspondre aux qualités publiées par les fournisseurs de services. Les utilisateurs finaux et les fournisseurs doivent alors s'entendre sur une même sémantique concernant les qualités et les valeurs de satisfaction, afin de faciliter le processus de sélection.

On est alors confronté à un problème de consensus sur la qualité de service. La communauté SOC (Ran,2003) (Maximilien&Singh,2004b) (Dobson et *al.*,2005) (Wang et *al.*,2006) (Giallonardo&Zimeo,2007) (Huang et *al.*, 2007) (Herssens et *al.*,2008) (Jureta et *al.*,2008) croit au besoin d'un partage du même vocabulaire qualité entre les fournisseurs et les clients de services. En effet, il existe des approches (Maximilien&Singh,2004b) (Dobson et *al.*,2005) (Wang et *al.*,2006) (Giallonardo&Zimeo,2007) (Huang et *al.*, 2007) qui proposent des ontologies de qualité en se basant sur la QoS des services Web, définie par le W3C (Lee et *al.*,2003), tandis que d'autres approches (Herssens et *al.*,2008) (Jureta et *al.*,2008) proposent d'étendre le modèle OMG UML QoS.

Quant à la communauté RE, elle est illustrée par des travaux comme ceux de Ma et ses collègues (Ma et *al.*,2009), qui proposent une ontologie de buts de qualité, laquelle se base sur le cadre NFR (*i.e. NFR Framework*) (Chung et *al.*,2000). Or, les solutions proposées par ces deux communautés demeurent déconnectés, voir isolées l'une de l'autre. La communauté SOC fournit des ontologies regroupant des métriques, qui sont de niveau opérationnel. Néanmoins, elles ne sont pas facilement exploitables par l'utilisateur final du service, qui souhaite utiliser cette ontologie pour exprimer ses exigences non-fonctionnelles. La communauté RE, quant à elle, propose une ontologie orientée but, laquelle reste déconnectée des mesures logicielles. Malgré tous les efforts de recherche réalisés jusqu'ici, il n'existe pas un vrai consensus, en matière de description de la qualité, entre utilisateurs finaux et fournisseurs de service, en lien avec cette discordance conceptuelle persistante. Cette absence de consensus affecte sensiblement le processus de sélection, car, d'un côté, les clients, lorsqu'il s'agit des utilisateurs finaux, ont des difficultés à exprimer leurs besoins à l'aide des outils offerts par les fournisseurs (la communauté SOC en général). D'un autre côté, les outils proposés par la communauté RE sont déconnectés des services logiciels, n'étant ainsi pas applicable à l'exécution d'un service.

Le problème de consensus qualité implique alors un besoin d'uniformiser le vocabulaire qualité (les qualités et leurs évaluations) entre les utilisateurs finaux et les fournisseurs de service d'un même domaine. Ce vocabulaire (les qualités et leurs évaluations) doit être compréhensible par l'utilisateur final, lequel peut exprimer ses exigences non-fonctionnelles

en se basant sur ce vocabulaire. Il doit également être utilisé par les fournisseurs de services pour décrire la qualité de leurs services de haut niveau et logiciel. Ce vocabulaire permet alors de diminuer les risques d'incompatibilité lors du processus de sélection de services, d'améliorer la crédibilité des fournisseurs, et de rapprocher les mesures quantitatives délivrées par les services logiciels aux évaluations qualitatives des services de haut niveau.

1.2.3 LA SELECTION DES SERVICES

La sélection de service est définie par Maximilien et Singh (Maximilien&Singh,2004b) comme étant l'étape permettant de différencier les services fournissant la même fonctionnalité, en se basant sur la qualité de service. La sélection de services est souvent réalisée pour faire de la composition, de la substitution ou de la configuration de service (O'Sullivan et *al.*,2002) (Orriëns et *al.*,2003).

Les approches appartenant à la communauté SOC utilisent des algorithmes de sélection, lesquels permettent dynamiquement (à l'exécution) de sélectionner les services implémentant une fonctionnalité donnée, en fonction de leurs QoS. Lorsqu'il s'agit d'une optimisation locale ou de la prise en compte de qualités conflictuelles (Baryannis et *al.,2008*) (Herssens et *al.*,2008), les méthodes multicritères d'aide à la prise de décision (Zeng et *al.*,2003) (Wang et *al.*,2006) (Herssens et *al.*,2008) sont les plus utilisées. Dans le cas d'une optimisation globale, celle-ci est considérée comme un problème NP-complet (Baryannis et *al.,2008*), lequel est résolu généralement par de la programmation linéaire (Zeng et *al.*,2003) ou des heuristiques (Jaeger et *al.*,2005) (Yu et *al.*,2007).

Quant aux approches de la communauté RE, elles considèrent les buts qualité comme les critères de sélection des services. Elles proposent de sélectionner, lors de la conception du système logiciel, les services qui d'une part répondent aux exigences fonctionnelles des utilisateurs finaux, et qui d'autre part ont le plus de similitudes avec leurs exigences non-fonctionnelles (Penserini et *al.*,2006) (Ma et *al.*,2009). Différentes techniques sont utilisées pour cela, dont les méthodes multicritères d'aide à la décision, proposées par Ma et ses collègues (Ma et *al.*,2009).

Nous remarquons que la sélection des services dans la communauté SOC est effectuée à l'exécution des services logiciels, tout en considérant leurs qualités de service. Toutefois, les services sélectionnés restent difficilement accessibles aux utilisateurs finaux, car ils correspondent à des descriptions techniques des services et de la qualité qu'ils offrent. Les auteurs de la communauté RE, quant à eux, font la sélection des services de haut niveau

durant la conception de système, en considérant les exigences non-fonctionnelles des utilisateurs finaux. Néanmoins, ils omettent la perspective logicielle de ces services.

Il est donc important d'adapter la sélection du niveau SOC au niveau métier, afin de combler le fossé entre la sélection de service de haut niveau et la sélection des services logiciels de bas niveau. Ainsi, le résultat de la sélection doit être compréhensible à l'utilisateur final, sans pour autant négliger la perspective logicielle des services de haut niveau.

On est alors confronté à un problème de sélection et d'exécution des services de haut niveau, en fonction des exigences non-fonctionnelles des utilisateurs finaux. Ce problème implique qu'il faut guider l'utilisateur final, lors de la sélection des services de haut niveau, en fonction des exigences non-fonctionnelles exprimées par cet utilisateur, tout en tenant compte de la perspective logicielle propre à ces services.

1.2.4 LA CONFIGURATION DES SERVICES

D'après Teije et ses collègues (Teije et *al.*,2004), la configuration de service consiste à adapter le service *Web* composite en fonction des besoins de chaque client potentiel (Teije et *al.*,2004).

Dans les travaux de la communauté SOC, Xiong et ses collègues (Xiong et *al.*,2008) proposent de définir formellement les dépendances fonctionnelles du service *Web* composite. Ces auteurs indiquent que plusieurs configurations de ce service peuvent répondre aux exigences fonctionnelles, mais offrir des QoS différentes. Ils proposent de définir un algorithme permettant de retourner la configuration optimale ayant les meilleurs paramètres de QoS.

Selon Teije et ses collègues (Teije et *al.*,2004), le service *Web* composite, une fois défini, peut très bien répondre aux besoins de quelques utilisateurs, comme il peut ne pas répondre aux exigences d'autres. Pour cela, ces auteurs proposent de considérer le service Web composite comme un modèle fixe paramétré, qui doit être configuré afin de répondre aux besoins de chaque client potentiel. Ainsi, les services sont modélisés en se basant sur le principe de généricité, afin qu'ils ne soient pas dédiés à un utilisateur particulier. D'après, (Crescenzoe et *al.*,2006), la généricité regroupe traditionnellement les problèmes de variabilité et ceux d'adaptation à un contexte spécifique. Récemment, Cohen et Krut (Cohen&Krut, 2010) proposent de se baser sur les approches de ligne de produit logiciel (*SPL-Software Product Lines*) pour gérer la variabilité dans le développement des systèmes-orienté services. Au dire de ces auteurs, cette variabilité est gérée par le processus de

configuration qui correspond à l'ensemble des variantes et leurs combinaisons significatives, qui répondent à des buts spécifiques.

Ultérieurement, Acher et ses collègues (Acher et *al.*,2010) parlent de configuration de *Workflows*, en assemblant des services paramétrés dans un *Workflow*. Ces auteurs parlent de famille de services comme étant l'ensemble de services qui exhibent de la variabilité et qui peuvent être représentés par les langages de modélisation de la variabilité (modèle de *Features* dans ce cas). Ils proposent alors d'organiser les services comme des éléments d'une ligne de produit, en mettant en évidence la variabilité exhibée par ces services. Aussi, Mohabbati et ses collègues (Mohabbati et *al.*,2011) proposent une approche de configuration de ligne de produit logiciel orienté service (*SOSP-Service-Oriented Software Product Lines*). Particulièrement, ils utilisent une méthode d'optimisation linéaire pour la sélection du service optimal, en fonction des exigences non-fonctionnelles.

Dans les approches de RE, le processus de configuration de produit consiste à dériver des produits individuels à partir des éléments d'une ligne de produits en figeant certains choix vis-à-vis des éléments variables, et en considérant les restrictions relatives à la compatibilité de ces éléments. Dans le cas de diverses configurations similaires aux exigences et aux restrictions, il faut alors effectuer un choix selon les critères de qualité. Les approches RE s'intéressent d'une part à la configuration de processus métier de haut niveau (Lapouchnian, et *al.*,2007), et d'autre part à celle des lignes de produit (Niemelä&Immonen,2007) (Rohleder,2009) (Djebbi,2011).

Dans les lignes de produit logiciel, la variabilité est définie de façon explicite (Clements&Northrop,2001). Par exemple, Ommering (Ommering,2002) considère la réutilisation entre les lignes de produit nécessaire, tandis que Thompson et Heimdahl (Thompson&Heimdahl,2002) observent que les parties variables peuvent être partagées à travers les lignes de produit. Ainsi, il existe plusieurs modèles de variabilité dont celui de Bühne et ses collègues (Bühne et *al.*,2005) qui proposent une modélisation explicite des informations de variabilité à travers les concepts de variantes et de points de variation dans les lignes de produits. Particulièrement, ils relèvent, comme (Lutz,2000), la définition des contraintes de variabilité correspondant aux contraintes de dépendance entre les exigences variables. Ces contraintes de dépendance sont importantes pour la validité de la ligne de produit. Rohleder (Rohleder,2009) propose une approche orientée but permettant de dériver une configuration de produit, en considérant la variabilité fonctionnelle et non-fonctionnelle des exigences.

Comme (Cohen&Krut,2010) (Mohabbati et *al.*,2011), nous pensons que la configuration de service s'apparente à la configuration de lignes de produit, afin de gérer la variation dans les services. De ce fait, et par analogie à la configuration de produit, nous pensons qu'il est important de faire une configuration exécutable (*i.e.* une configuration valide pouvant être exécutée) des services de haut niveau. Celle-ci consiste à composer des services élémentaires, en définissant des choix par rapport aux variantes des services, tout en considérant les restrictions concernant la compatibilité de ces services.

On est alors confronté à un problème de configurations et d'exécution des services de haut niveau, de façon à gérer la variabilité métier. Ce problème implique qu'il faut guider l'utilisateur final, lors de la configuration des services de haut niveau, tout en considérant les contraintes de dépendance entre ces services, et les exigences non-fonctionnelles exprimées par cet utilisateur.

1.3 *RESULTATS*

Afin d'apporter une solution aux différents problèmes énoncés plus haut, nous nous positionnons dans la perspective de l'utilisateur final du service tout en considérant aussi bien les services de haut niveau, que les services logiciels de bas niveau qui les opérationnalisent. Nous adoptons dans cette thèse une démarche de l'ingénierie des exigences (*RE-Requirements Engineering*), sans omettre la vision technique des services promus par la communauté SOC. Par ailleurs, nous focalisons sur les services dédiés à l'utilisateur, car ils sont particulièrement confrontés aux problèmes énumérés ci-dessus, du fait qu'ils interagissent directement avec les utilisateurs finaux.

Pour résoudre le problème de discordance conceptuelle entre les exigences fonctionnelles des utilisateurs finaux et les services logiciels, nous adoptons l'architecture intentionnelle orientée service (*iSOA-intentional Service Oriented Architecture*) suggérée par Kaabi, Rolland et leurs collègues (Kaabi,2007) (Rolland et *al.*,2007) (Rolland et *al.*,2008). L'architecture iSOA permet de déplacer la vision orientée fonction du service vers une vision orientée intention. Le *service intentionnel*, définie dans l'architecture iSOA, permet de décrire le service en faisant référence aux buts et aux stratégies que l'organisation choisit pour les atteindre. Par conséquent, les interfaces de ces services mettent en évidence le but métier que le service permet de réaliser, plutôt que la description technique. Ainsi, les fournisseurs métier décrivent les services dans un mode intentionnel et les publient dans un annuaire de services intentionnels. Les agents métier (au lieu des agents logiciels) sont des utilisateurs finaux, lesquels effectuent une découverte, dirigée par les intentions, d'un ou plusieurs services

correspondant à ses exigences fonctionnelles. Par ailleurs, le service intentionnel peut être atomique ou agrégat. Un *service atomique* est associé à un but opérationnalisable, lequel supporte l'opérationnalisation du service intentionnel vers les *services logiciels*. Un *service agrégat* cherche à satisfaire un but tactique ou stratégique par la composition de différents services composants (atomiques ou agrégats). Cette composition permet non seulement de fournir des services de plus grosses granularités, mais également de décrire les différentes variantes à l'intérieur de ce service. En d'autres termes, le service intentionnel agrégat introduit de la variabilité, en décrivant les différentes variantes permettant de réaliser le but du service. Ainsi, le service intentionnel peut s'adapter au contexte métier des agents métier.

Néanmoins, dans iSOA, la description actuelle du service intentionnel reste insuffisante. Le service intentionnel correspond aux exigences fonctionnelles des agents métier : il décrit le but fonctionnel que le service permet de réaliser (Rolland&Kaabi,2007). Néanmoins, iSOA reste insuffisante concernant la qualité de service et les contraintes de dépendance.

Ainsi, nous proposons une extension de l'architecture iSOA afin d'apporter une solution aux différents problèmes énoncés plus haut :

(1) Pour résoudre les problèmes de discordance conceptuelle entre les exigences non-fonctionnelles des utilisateurs finaux et la qualité des services logiciels, celui des dépendances fonctionnelles entre les services, nous proposons d'étendre l'architecture iSOA à travers un nouveau modèle nommé le *modèle intentionnel de la qualité de service*, dit *MiS-q*. Le modèle intentionnel à variation et qualitatif *MiS-q* remplace alors le modèle actuel *MiS*, dans l'architecture iSOA. Les fournisseurs métiers, via le modèle *MiS-q*, associent plusieurs éléments à leurs services :

   a. la variabilité : le modèle *MiS-q* permet de représenter la variabilité métier (existait déjà dans le modèle *MiS*) du service intentionnel. Cette variabilité correspond aux différentes alternatives de réalisation du but associé au service intentionnel. Ainsi, des points de variation sont définis à travers le service à variation. Le modèle *MiS-q* et la démarche de publication permet de concrétiser les intentions par des services logiciels.

   b. une dimension qualitative : nous proposons, via ce même modèle, la notion de *qualité du service intentionnel (QoiS-Quality of intentional Service*). La QoiS peut être une QoiS simple ou une QoiS globale. La QoiS simple est associée au service atomique, tandis que la QoiS globale correspond au service agrégat. La QoiS simple est définie comme étant l'ensemble des

buts qualité que le service atomique permet partiellement de satisfaire. L'évaluation de la QoiS est orientée but. Le modèle *MiS-q* permet ainsi d'établir la mise en correspondance au niveau intentionnel entre les exigences non-fonctionnelles des agents métier et la qualité des services logiciels.

c. des contraintes de dépendance entre les services : nous proposons, grâce au modèle *MiS-q*, de tenir compte de la variabilité métier du service intentionnel et de définir des *contraintes de dépendances* entre les services, qui permettent de préciser que le choix d'une variante peut ainsi entrainer le choix (ou l'exclusion) d'une autre variante. Ces dépendances permettent de définir des restrictions concernant la compatibilité de ces variantes, et de diriger ainsi la configuration du service agrégat, en garantissant sa cohérence métier au niveau exécution.

(2) Pour résoudre le problème de consensus sur la qualité, nous proposons un méta-modèle de la qualité, nommé *référentiel qualité*. Le référentiel qualité se base sur les concepts du cadre NFR *(NFR framework*) (Chung et *al.*,2000). Ce cadre permet de traiter les exigences non-fonctionnelles, telles que la sécurité, la performance et la précision, comme des buts qualité à partiellement satisfaire. Nous proposons d'augmenter le cadre d'application NFR, en associant aux buts qualité des métriques permettant de les quantifier. A travers ces métriques, le référentiel qualité permet d'associer à des seuils de satisfaction prédéfinis, une évaluation quantitative moins subjective et qui peut être directement reliée aux services logiciels. Nous proposons que les fournisseurs métier utilisent le référentiel qualité pour décrire la qualité de leurs services. De façon identique, les agents métier décrivent leurs exigences non-fonctionnelles dans un *contexte qualité*, en se basant sur ce même référentiel qualité. Le référentiel qualité permet d'alimenter le modèle *MiS-q* et celui du contexte qualité, de façon unifiée, afin de diminuer le risque d'incompatibilité lors de la sélection des services. Il permet également d'améliorer la crédibilité des fournisseurs par la définition des seuils de satisfaction en fonction des métriques quantitatives.

(3) Pour résoudre le problème de sélection des services de haut niveau, à l'exécution, en fonction des exigences non-fonctionnelles des utilisateurs finaux, nous proposons une sélection intentionnelle des services. La sélection de services dans l'architecture iSOA consiste à choisir, parmi les services des différents fournisseurs, ceux qui correspondent au mieux au contexte qualité de l'agent métier. En effet, plusieurs fournisseurs métier

peuvent fournir des services similaires, c'est-à-dire réalisant le même but fonctionnel. La découverte de services, en fonction du but fonctionnel, n'est pas prise en compte dans cette thèse. Néanmoins, ces services offrent potentiellement des QoiS différentes. Ils permettent de satisfaire différents buts qualité avec des seuils de satisfaction également différents, voir ne satisfaire aucun but qualité. Nous proposons de faire une sélection intentionnelle des services, en adaptant la méthode multicritères d'aide à la décision TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) (Hwang&Yoon,1981) (Zavadskas,1986). Par ailleurs, un outillage est proposé pour démontrer la faisabilité de la méthode de sélection intentionnelle des services que nous proposons.

(4) Pour résoudre le problème de configuration exécutable des services de haut niveau, dirigée par les exigences non-fonctionnelles des utilisateurs finaux, nous proposons une configuration intentionnelle des services. La configuration de service dans l'architecture iSOA est dirigée par les intentions. En effet, les variantes de service sont choisies uniquement en fonction des aspects fonctionnels. La configuration est étendue pour prendre en compte les aspects fonctionnels et non-fonctionnels demandés par l'utilisateur. La méthode multicritères d'aide à la décision TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) (Hwang&Yoon,1981) (Zavadskas,1986) est utilisée pour sélectionner les variantes métier qui offrent la meilleure qualité, en fonction des exigences non-fonctionnelles de l'utilisateur. De plus, il faut s'assurer que le résultat obtenu est exécutable ou cohérent au niveau métier. Pour cela, l'algorithme de configuration doit prendre en compte les contraintes de dépendances exprimées dans le modèle *MiS-q.*

1.4 ***PLAN***

Le présent rapport de thèse est organisé en plusieurs chapitres de façon à faciliter et à guider le lecteur dans la compréhension de notre travail.

Le chapitre 2 présente une analyse de l'état de l'art, en utilisant un cadre de référence, lequel permet de comparer sept approches de l'ingénierie des applications à base de services traitant de la qualité de services.

Le chapitre 3 présente un aperçu de la solution que nous proposons, concernant la sélection intentionnelle des services, en fonction des exigences non-fonctionnelles des clients.

Le chapitre 4 présente l'architecture intentionnelle des services (*iSOA-intentional Service Oriented Architecture*).

Le chapitre 5 présente le méta-modèle du référentiel qualité, lequel permet d'améliorer le consensus qualité entre les fournisseurs et les agents métier.

Le chapitre 6 présente les différents concepts du méta-modèle intentionnel de la qualité de service, *MiS-q*, lequel contribue à réduire la discordance conceptuelle. Le méta-modèle *MiS-q* est une extension du modèle *MiS* (Kaabi,2007), qui met l'accent sur deux éléments importants, à savoir : (i) la qualité du service intentionnel (*QoiS-Quality of intentional Service)*, ainsi que les différents types de QoiS, dans une perspective dirigée par les buts ; et (ii) les contraintes de dépendance entre les services, permettant une configuration exécutable du service agrégat.

Le chapitre 7 décrit la méthodologie permettant l'obtention du modèle *MiS-q.* Celle-ci définit l'intégration de la qualité au service intentionnel, en se basant sur le référentiel qualité.

Le chapitre 8 présente le processus de sélection et de configuration intentionnelle des services.

Le chapitre 9 présente le prototype de sélection (et de configuration) intentionnelle des services.

Le chapitre 10 illustre l'approche proposée par un exemple.

Le chapitre 11 conclut ce mémoire de thèse en résumant la plus-value de ce travail et en proposant des perspectives de recherche.

CHAPITRE 2 : ETAT DE L'ART

2.1 *INTRODUCTION*

Ce chapitre est consacré à l'état de l'art du domaine de l'ingénierie des systèmes à base de services. Nous nous intéressons plus particulièrement aux différentes approches qui traitent de la qualité de service (*QoS- Quality of Service*) en vue de la sélection et de la configuration des services.

Ce chapitre a pour but de définir un cadre de référence multidimensionnel permettant de comparer plusieurs travaux, lesquels traitent de la QoS. Ce cadre de référence, inspiré de (Rolland et *al.*,1998a), est composé de quatre vues. Chacune des vues explore des caractéristiques particulières de la qualité de service. Ainsi, ce cadre permet de positionner différentes approches de recherche les unes par rapport aux autres.

Toutefois, avant de présenter le cadre de référence, nous proposons un ensemble de définitions reliées au domaine de l'ingénierie des systèmes à base de services, particulièrement les systèmes qui prennent en compte la qualité de service.

Le présent chapitre est organisé comme suit. La section 2 consiste à présenter les différents concepts relatifs à la qualité de service et aux exigences non-fonctionnelles. La section 3 présente un cadre de référence de la qualité de service, permettant d'évaluer différentes approches. La section 4 propose le positionnement de six approches, en fonction du cadre de référence. La section 5 clôture ce chapitre par un résumé de l'évaluation.

2.2 *PRESENTATION DES CONCEPTS*

La croissance d'Internet a engendré la refonte des systèmes d'information des organisations et a entraîné une réorganisation de la façon dont les entreprises interagissent avec leurs partenaires et clients. Cette interaction est désormais construite autour de la notion de service.

2.2.1 LA NOTION DE SERVICE

Le besoin d'interopérabilité entre applications hétérogènes, développées sur des plates-formes différentes a renforcé l'émergence du paradigme informatique orienté service (*SOC-Service Oriented Computing)* (Papazoglou&Georgakapoulos,2003) (Alonso et *al.*,2004) (Huhns&Singh,2005). Le paradigme SOC se base sur la notion de service, laquelle est

matérialisée par un composant logiciel assurant une fonctionnalité particulière et accessible via son interface.

Les services sont une suite logique des composants logiciels et des approches d'intégration d'applications (Pfister&Szyperski,1996). Les composants sont des entités logicielles indépendantes utilisant des interfaces pour communiquer au sein d'une même application ou en composants distribués (Brown,1996) (Szyperski,1997). Cependant, l'architecture à base de composants, bien qu'utilisant un modèle objet distribué, impose sa propre infrastructure et une liaison forte entre les fonctionnalités offertes par les composants et leurs clients. Les applications construites à base de cette architecture ne permettent pas la communication entre clients hétérogènes.

Plusieurs définitions de la notion de service existent dans la littérature. Nous retenons celle de Papazoglou et ses collègues (Papazoglou et *al.*,2008) qui définissent les services comme des entités logicielles autonomes, indépendantes des plateformes et pouvant être utilisées de façon indépendante des plateformes. Les services sont utilisés comme des briques logicielles afin de supporter un développement rapide, à moindre coût et une composition facile d'applications distribuées dans des environnements hétérogènes (Papazoglou et *al.*,2008).

Un service est une fonction métier bien définie et autonome qui ne dépend pas de l'état ou du contexte des autres services, d'où sa caractéristique de faible couplage. La technologie utilisée pour fournir le service, tel que le langage de programmation, ne fait pas partie de sa définition. Le service est indépendant de la plateforme et reste donc interopérable. L'absence de dépendance entre services et leurs consommateurs permet d'ordonner les services dans de nombreux flux réalisant des différents processus métiers (Fremantle et *al.*,2002) (Papazoglou et *al.*,2008).

Les services Web sont actuellement la technologie la plus prometteuse qui se base sur le principe de SOC (Weerawarana et *al.*,2005). Le W3C (*World Wide Web Consortium*) (W3C,2004) définit le service web comme un système logiciel identifié par une adresse URL *(Uniform Resource Identifier),* dont les interfaces sont définies et décrites en utilisant XML. Sa définition peut être découvert par d'autres systèmes logiciels. Ces systèmes peuvent alors interagir avec le service Web dans une manière prescrite par sa définition, en utilisant des messages basés sur XML et véhiculés par des protocoles Internet.

La croissance du Web révolutionne la façon dont les entreprises interagissent avec leurs partenaires et leurs clients. Les services sont de plus en plus dédiés à des utilisateurs humains, plutôt qu'à des applications. Selon (Pallos,2001), un service métier est un groupement logique de composants requis pour satisfaire une demande métier

particulière. La notion de service a alors évolué grâce à la prolifération du commerce électronique, à travers les applications *Business-to-client* (B2C) (*e.g.* suivi du trafic) et les *Business-to-Business* (B2B) (*e.g.* facturation) (Casati&Shan,2001) (Medjahed et *al.*,2003). Le service est alors dit « e-service », lequel est décrit comme un service métier (*i.e.* fonctions métier) exposé par une entreprise sur Internet afin de fournir un moyen d'utiliser ce service à distance (Piccinelli et *al.*,2003). Les e-services peuvent être utilisés par des personnes (*business to consumer*), entreprises (*business to business*) et des appareils électroniques. C'est donc un terme plus large que les notions de commerce électronique ou de *e-business* (Dustdar&Schreiner,2005).

Aussi, l'utilisateur final peut avoir des difficultés d'une part à exprimer ses besoins, et d'autre part à exploiter ces services, car il doit faire face à des notions techniques qu'il ne maitrise pas forcément. Pour cela, plusieurs approches de l'ingénierie des exigences (*RE-Requirements Engineering*) (Guzélian,2007) (Mirbel&Crescenzo,2009) (Aiello&Giorgini,2004) (Penserini&Mylopoulos,2005) (Da Silva Santos et *al.*,2009) (Rolland et *al.*,2008) proposent d'utiliser la notion de but pour exprimer ce que l'utilisateur souhaite voir réalisé par le service. L'utilisation du but vise à hisser la définition des exigences des clients de service et celle du service à un haut niveau d'abstraction, facilitant ainsi son utilisation et sa compréhension par l'utilisateur final.

Le principe clé du paradigme SOC est l'architecture orientée services (*SOA-Service-Oriented Architecture*) (Alonso et *al.*,2004).

2.2.2 L'ARCHITECTURE ORIENTEE SERVICE

L'architecture orientée services (*SOA-Service Oriented Architecture*) est le moyen logique de concevoir des systèmes logiciels à base de service (Papazoglou et *al.*,2008). L'objectif de cette architecture est d'adresser les exigences de faible couplage et d'interopérabilité en fournissant, aux utilisateurs finaux ou à d'autres services des services publiés et découvrables via des interfaces précises (Alonso et *al.*,2004). L'idée maîtresse de l'architecture orientée service est que tout élément du système d'information doit devenir un service identifiable, documenté, fiable, indépendant des autres services, accessible, et réalisant un ensemble de tâches parfaitement définies (Booth et *al.*,2004).

La Figure 3 présente le modèle SOA, lequel est composé de trois acteurs principaux : le fournisseur de service, le client et l'annuaire des services (O' Sullivan, et *al.*,2002). Les interactions entre ces participants correspondent au cycle de vie d'un service : le fournisseur permet l'accès à son service à travers une interface. Le client désigne une personne, un

serveur ou une autre application qui accède au service et l'invoque à travers son interface. L'annuaire joue le rôle d'intermédiaire entre le fournisseur et le client. Les fournisseurs y enregistrent leurs services, et les clients y cherchent le service satisfaisant leurs besoins.

Le modèle de fonctionnement de SOA est supporté par une infrastructure technologique utilisant des standards permettant de publier, de découvrir et d'invoquer les services (Chauvet,2002)(Papazoglou,2007). Par exemple, L'infrastructure des services web s'est concrétisée autour de trois spécifications considérées comme des standards, à savoir WSDL, UDDI et SOAP (Cerami,2002). Le WSDL *(Web Service Description Language*) (Wsdl,2001) est utilisé pour la description des services ; le UDDI (*Universal Description, Discovery, and Integration*) (Uddi,2004) est adopté pour leur publication ; et le SOAP (*Simple Object Access Protocol*) (Soap,2007) est utilisé pour l'échange de messages. SOAP utilise HTTP (*HyperText Transfert Protocol*) (Http,2009) pour transporter des messages SOAP et XML (*Extensible Markup Language*) pour structurer les données.

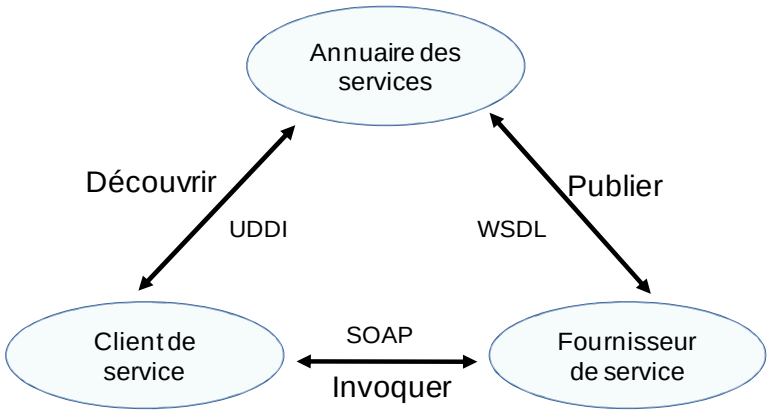

Figure 3. Le modèle SOA (O' Sullivan et *al.*,2002)

2.2.3 LA COMPOSITION DE SERVICES

Dans le cas où les fonctionnalités offertes par les services élémentaires sont pauvres et ne peuvent satisfaire les requêtes des utilisateurs, le processus de composition de services (*cf.* Figure 4) s'avère l'approche adéquate permettant de fédérer des services élémentaires dans des services composites ayant une valeur ajoutée et une fonctionnalité plus riche (Peltz,2003) (Papazoglou&Georgakapoulos,2003) (Papazoglou,2005) (Alonso et *al.*,2004) (Dustdar&Schreiner,2005) (Benatallah et *al.*,2005). La composition de service implique la capacité de sélectionner, de composer et de faire inter-opérer des services existants (Kellert&Toumani,2004). C'est une approche prometteuse supportant l'intégration des applications métier à l'intérieur et à l'extérieur des organisations. Elle permet, comme illustré à la Figure 4, de fédérer des services (atomiques ou composites) dans des services composites,

dont la logique métier exprimée correspond à un modèle de processus (Benatallah et *al.*,2002) (Papazoglou et *al.*,2008). Pour intégrer véritablement des processus métier à travers les frontières des entreprises, il est nécessaire de supporter des interactions métier de longue durée, lesquelles sont dirigées par un modèle de processus explicite (Van der Aalst,2003). Cela nécessite des langages de composition de services *Web*, tels que BPEL (*Business Process Execution Language*) (Andrews et *al.*,2003), *WS-Choreography* (Peltz,2003) et *WS-CDL* (*Web Services Choreography Description Language*) (Mendling&Hafner,2008).

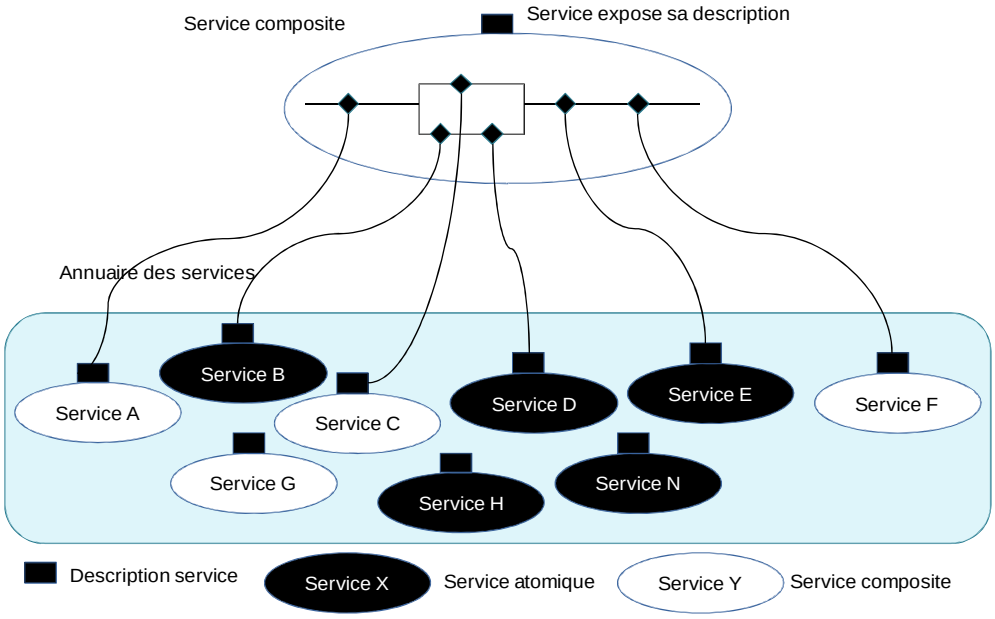

Figure 4. Représentation d'un service composite (OMG,2008)

Néanmoins, de tels langages ne répondent pas à la composition dynamique des services, ni à la configuration dynamique des processus métiers (Orriëns et *al.*,2003), car le nombre de services disponibles évolue très rapidement et la multitude de fournisseurs de services oblige les entreprises à adapter leurs services pour mieux répondre aux besoins des clients et ce à moindre coût. La composition dynamique de service consiste à sélectionner durant l'exécution du service composite, les services qui s'adapte aux exigences des utilisateurs, alors que la configuration des services consiste à adapter un service composite en fonction des besoins des clients.

Avec la croissance de la popularité de SOC, les systèmes informatiques cherchent désormais à se rapprocher des gestionnaires en adaptant le paradigme SOA au monde des entreprises et des processus métier qu'elles supportent. L'objectif étant d'automatiser, par composition de services, les processus métier inter et intra-entreprises, tels que les ventes, les chaînes de production, *etc*. Néanmoins, pour que le paradigme SOC soit transposé au niveau du métier de l'entreprise, il est nécessaire d'établir un lien entre les services de haut niveau,

lesquels sont compréhensibles par le monde du métier, et les services techniques et logiciels de bas niveau, développés par les fournisseurs de services (Azsanjani,2004) (Zimmermann et *al.*,2004) (Rolland et *al.*,2008) (Da Silva Santos et *al.*,2009). Pour cela, plusieurs approches de l'ingénierie des exigences (*RE-Requirements Engineering*) (Penserini&Mylopoulos,2005) (Aiello&Giorgini,2004) (Stollberg&Norton,2007) (Da Silva Santos et *al.*,2009) (Rolland et *al.*,2008) proposent d'utiliser les buts pour se rapprocher de l'expression des besoins des utilisateurs métier. Quelques approches (Penserini&Mylopoulos,2005) (Aiello&Giorgini,2004) modélisent les services comme des tâches réalisant un but métier, dans l'optique de sélection de service, sans autant s'intéresser à la perspective logicielle de ces services. D'autres approches (Stollberg&Norton,2007) (Da Silva Santos et *al.*,2009) portent sur la spécification de buts dans l'optique de recherche de services Web. Ces approches ne s'attachent pas au problème de capture de ces buts. Rolland et ses collègues (Rolland et *al.*,2008) suggèrent de déplacer la vision orientée-fonction de SOC vers une vision orientée-intention grâce à l'architecture orientée service intentionnel (*iSOA-intentional Service Oriented Architecture*). La capture des exigences fonctionnelles des utilisateurs est effectuée en utilisant un modèle de but. Les services, dits services intentionnels, sont identifiés à partir de ce modèle de but. La publication, la recherche, la composition et l'exécution (par des services logiciels) des services intentionnels se font sur la base de ces descriptions intentionnelles. Le service intentionnel répond aux exigences fonctionnelles des clients, sans autant considérer leurs exigences non-fonctionnelles, lesquelles correspondent à la qualité fournie par les services.

2.2.4 LA QUALITE DE SERVICE

Généralement, un service est défini en termes de son comportement fonctionnel, c'est-à-dire de ce que le service fait ou réalise comme fonctionnalité. Cependant, cette définition ne décrit pas entièrement le comportement du service. Ainsi, O' Sullivan et ses collègues (O' Sullivan et *al.*,2002) considèrent que la description d'un service n'est complète que lorsque ses qualités sont exprimées. Particulièrement, la qualité de service est importante pour sélectionner les services devant faire partie d'une composition de services (O' Sullivan et *al.*,2002).

La Norme ISO (ISO 9000,2000) (ISO/IEC 9001,2008) décrit la qualité comme l'aptitude d'un ensemble de caractéristiques intrinsèques à satisfaire des exigences. La qualité de service est définie par Schmidt et ses collègues comme tout attribut, lequel n'est pas un comportement fonctionnel du système (Schmidt et *al.*,1998). Siqueira et Cahill (Siqueira&Cahill,1998) utilisent la qualité de service pour désigner les exigences imposées

par l'utilisateur ou un composant du système vis-à-vis de la performance d'une application, durant son exécution.

2.2.4.1 *LA QUALITE DE SERVICE DANS SOC*

La qualité de service joue un rôle prépondérant dans la distinction des multitudes de services Web fournissant la même fonctionnalité.

2.2.4.1.1 DEFINITION DE LA QUALITE DE SERVICE

Dans la communauté SOC, le concept de qualité de service (*QoS- Quality of Service*) est originaire de la communauté des réseaux et celle des télécommunications (Clark et *al.*,1992) (ITU,1994) (Cruz,1995) (FIPA,2002) (Tian et *al.*,2003), lesquelles considèrent la qualité comme les paramètres liés aux caractéristiques du réseau, tels que la bande passante, la latence, la sécurité et la disponibilité du réseau. La QoS se réfère, en plus des caractéristiques du réseau, aux caractéristiques du fournisseur, du client et de l'implémentation du service (Mani&Nagarajian,2002) (Dobson et a*l.*,2005).

Communément, la qualité de service se réfère aux propriétés non-fonctionnelles des services Web, telles que la performance, la fiabilité et les exigences reliées au réseau (Ran,2003) (Sumra&Arulazi,2003) (W3C,2003), ainsi que toute autre qualité définie par l'utilisateur (Singh&Bilgin,2004). Dobson et ses collègues (Dobson et *al.*,2005) différencient la QoS spécifique de la QoS générique. La QoS spécifique est en relation avec une logique métier particulière, tandis que la QoS générique est partagée entre les attributs mesurables, tels que le temps de réponse (*i.e.* temps requis pour compléter une requête du service web) et la fiabilité (*i.e.* capacité d'un service d'exécuter correctement ses fonctions), les attributs non mesurables, tels que le prix d'exécution (*i.e.* le prix qu'un client du service doit payer pour bénéficier du service) et la sécurité (*i.e.* cryptage des messages et le contrôle d'accès).

Fedosseev (Fedosseev,2004) considère la qualité de service d'un processus métier (*i.e.* une composition de service) comme l'ensemble des caractéristiques quantitatives et qualitatives, nécessaires à la réalisation d'un ensemble d'exigences de départ. Les caractéristiques quantitatives, dites mesurables, sont définies par les métriques, telles que le coût et le temps de réponse. Les caractéristiques qualitatives, dites non mesurables, spécifient les services offerts par les systèmes, tels que les mécanismes de sécurité et de tolérance aux pannes.

2.2.4.1.2 MODELE DE QUALITE DE SERVICE

Plusieurs efforts ont été effectués pour évaluer la qualité d'un logiciel. Nous citons le modèle de qualité proposé par Boehm et ses collègues (Boehm et *al.*,1976) et celui de McCall

et Matsumoto (McCall&Matsumoto,1980). De ce dernier dérive le modèle de l'organisation internationale de normalisation ISO/IEC[1] 9126 (ISO/IEC 9126,2001), qui reste le plus connu. En effet, La norme ISO/IEC 9126 définit un langage commun pour modéliser la qualité d'un logiciel. Elle est partagée en quatre parties, à savoir le modèle de qualité, les métriques internes (*ang. Internal Metrics*), les métriques externes (*ang. External Metrics*) et les métriques en utilisation (*ang. Quality in Use Metrics*). Le modèle de qualité logicielle permet de classer de façon arborescente et structurée, sur la base de définitions standardisées, plusieurs dizaines de caractéristiques. Six *caractéristiques* sont définies par ISO/IEC 9126. Celles-ci sont elles-mêmes découpées en *sous-caractéristiques* de qualité (27 sous-caractéristiques dans la version la plus récente de la norme). Les caractéristiques de ISO/IEC 9126 sont (*cf.* Figure 5):

- Capacité fonctionnelle : *Est-ce que le logiciel répond aux besoins fonctionnels exprimés ?* Cette caractéristique est décomposée en Pertinence ; Exactitude ; Interopérabilité ; Sécurité ; et Conformité.
- Fiabilité : *Est-ce que le logiciel maintient son niveau de service dans des conditions précises et pendant une période déterminée ?* Cette caractéristique est subdivisée en Maturité ; Tolérance aux pannes ; Facilité de récupération ; et Conformité.
- Facilité d'utilisation : *Est-ce que le logiciel requiert peu d'effort à l'utilisation ?* Cette caractéristique est décomposée en Facilité de compréhension ; Facilité d'apprentissage ; Facilité d'exploitation ; Pouvoir d'attraction ; et Conformité.
- Rendement / Efficacité : *Est-ce que le logiciel requiert un dimensionnement rentable et proportionné de la plate-forme d'hébergement en regard des autres exigences ?* Cette caractéristique est subdivisée en Comportement temporel ; Utilisation des ressources ; et Conformité.
- Maintenabilité : *Est-ce que le logiciel requiert peu d'effort à son évolution par rapport aux nouveaux besoins ?* La Maintenabilité est décomposée en Facilité d'analyse ; Facilité de modification ; Stabilité ; Testabilité ; et Conformité.
- Portabilité : *Est-ce que le logiciel peut être transféré d'une plate-forme ou d'un environnement à un autre ?* La Portabilité caractéristique est subdivisée en Facilité d'adaptation ; Facilité d'installation ; Coexistence ; Interchangeabilité ; et Conformité.

[1] ISO (the International Organization for Standardization) et IEC (the International Electrotechnical Commission)

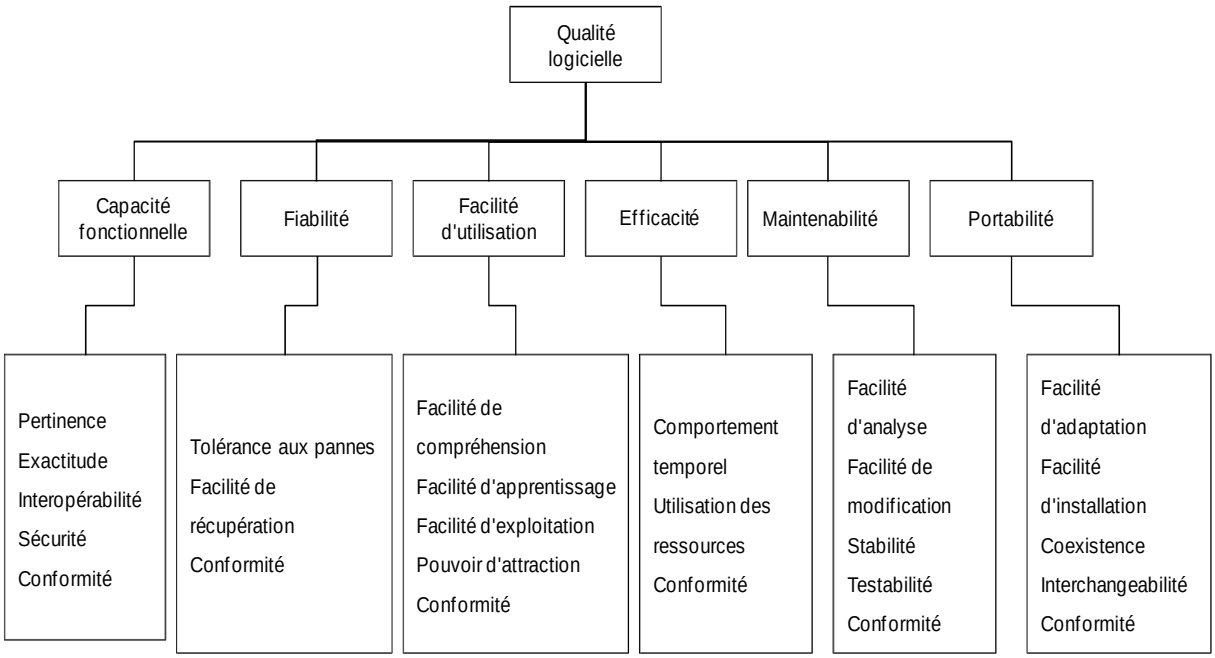

Figure 5. Qualité logicielle d'ISO/IEC 9126 (ISO/IEC 9126,2001)

Les métriques internes sont celles qui ne sont pas liées à l'exécution du logiciel (*i.e.* métrique statique), tandis que les métriques externes sont appliquées à l'exécution du logiciel. Enfin, les métriques en utilisation sont disponibles lorsque le produit final est utilisé dans ses conditions réelles. Elles correspondent à la capacité du produit à permettre aux utilisateurs de réaliser leurs buts (*e.g.* sureté, efficacité, *etc*.).

Le contenu de la norme ISO/IEC 9126 est repris, avec des enrichissements, par la norme ISO 25000, également appelée SQuaRE (pour *Software QUAlity Requirements and Evaluation*, exigences et évaluation de la qualité du logiciel) (ISO/IEC 25000,2005) (ISO/IEC 25010,2011).

Le groupe de travail Architecture des Services Web du W3C, lequel travaille sur les architectures des services web, a identifié et décrit un ensemble d'exigences QoS pour les services web (*ang. QoS requirements for web services*) (Lee et *al.*,2003), à savoir : la performance (qui englobe le débit (*ang.* throughput), le temps de réponse et le temps d'exécution), la fiabilité, la scalabilité ou la montée en échelle (*ang.* scalability), la capacité, la robustesse, le traitement d'exception, l'exactitude, l'intégrité, l'accessibilité, la disponibilité, l'interopérabilité, la sécurité, et les exigences en QoS liées au réseau.

L'OMG[2] propose un modèle de référence pour représenter la qualité de service, à savoir *OMG UML QoS* (OMG,2008). Le modèle OMG UML QoS permet de modéliser la qualité de

[2]Object Management Group

service et les mécanismes de tolérance aux pannes. Ce modèle regroupe plusieurs concepts associés à la qualité de service, tels que *QoS Characteristics*, *QoS Categories* et *QoS Value*. La qualité de service (*QoS Characteristics*) est définie comme des critères mesurables associés aux services, tels que le temps de réponse et la latence, et elle possède des valeurs (*QoS Value*). Elle est regroupée en une qualité abstraite (*QoS Categories*), telle que la performance et la sécurité.

Il n'existe pas de consensus bien précis au sujet de l'ensemble des QoS associées aux services web, mais la plupart des travaux de recherche (Ran,2003) (Araban&Sterling,2004) qui ont essayé d'identifier et de classifier les exigences QoS des services Web se basent sur celles définies par le W3C auxquelles sont souvent associées d'autres exigences. De même, il n'existe pas de consensus sur la modélisation de la qualité de service. Les approchent se partagent entre celles qui proposent des ontologies (Maximilien&Singh,2004b) (Dobson et *al.*,2005) (Afandi et *al.*,2006) (Giallonardo&Zimeo,2007). Herssens, Jureta et leurs collègues (Herssens et *al.*,2008) (Jureta et *al.*,2008) étendent le modèle OMG UML QoS pour faire de la sélection de services.

2.2.4.1.3 USAGE DE LA QUALITE DE SERVICE

Les approches de la communauté SOC (Tian et *al.*,2003) (Zeng et *al.*,2003) (Maximilien&Singh,2004b) (Dobson et *al.*,2005) (Papaioannou et *al.*,2006) (Afandi et *al.*,2006) (Herssens et *al.*,2008) utilisent la qualité de service dans l'optique de sélection des services Web. La sélection de services est définie par Maximilien et Singh (Maximilien&Singh,2004a) comme le moyen permettant de différencier les services fournissant la même fonctionnalité, en se basant sur la qualité de service. La sélection de services se fait dynamiquement, à l'exécution, afin de sélectionner les services implémentant une fonctionnalité donnée, en fonction de leurs QoS. La majorité de ces approches (Maximilien&Singh,2004b) (Dobson et *al.*,2005) (Papaioannou et *al.*,2006) (Afandi et *al.*,2006) modélise la qualité de services sous forme d'ontologies de qualité. L'objectif de ces ontologies est d'effectuer une correspondance sémantique entre les QoS publiées par les fournisseurs et les préférences QoS des clients. Lorsqu'il s'agit d'une optimisation locale (Baryannis et *al.*,*2008*) ou de la prise en compte de qualités conflictuelles (Herssens et *al.*,2008), la sélection des services est effectuée en utilisant les méthodes multicritères d'aide à la prise de décision (Zeng et *al.*,2003) (Wang et *al.*,2006) (Herssens et *al.*,2008). Dans le cas d'une optimisation globale, celle-ci est considérée comme un problème NP-complet

(Baryannis et *al.,2008*), lequel est résolu généralement par de la programmation linéaire (Zeng et *al.*,2003) ou des heuristiques (Jaeger et *al.*,2005) (Yu et *al.*,2007).

Teije et ses collègues (Teije et *al.*,2004) proposent de configurer le service composite. En effet, une fois celui-ci défini, il peut très bien répondre aux besoins de quelques utilisateurs, comme il peut ne pas répondre aux exigences d'autres. La configuration des services consiste alors à adapter le service composite en fonction des besoins de chaque client potentiel (Teije et *al.*,2004). Pour cela, ces auteurs proposent de considérer le service Web composite comme un modèle fixe paramétré, qui doit être configuré afin de répondre aux besoins de chaque client. Xiong et ses collègues (Xiong et *al.*,2008) proposent de définir formellement les dépendances fonctionnelles du service Web composite. Ces auteurs indiquent que plusieurs configurations de ce service peuvent répondre aux exigences fonctionnelles, mais offrir des QoS différentes. Ils proposent de définir un algorithme permettant de retourner la configuration optimale ayant les meilleurs paramètres de QoS.

Les services sont alors modélisés en se basant sur le principe de généricité, afin qu'ils ne soient pas dédiés à un utilisateur particulier. D'après, (Crescenzoe et *al.*,2006), la généricité regroupe traditionnellement les problèmes de variabilité et ceux d'adaptation à un contexte spécifique. Récemment, Cohen et Krut (Cohen&Krut,2010) proposent de se baser sur les approches de ligne de produit logiciel (*SPL-Software Product Lines*) pour gérer la variabilité dans le développement des systèmes-orienté services. Au dire de ces auteurs, cette variabilité est gérée par le processus de configuration qui correspond à l'ensemble des variantes et leurs combinaisons significatives, qui répondent à des buts spécifiques.

Ultérieurement, Acher et ses collègues (Acher et *al.*,2010) parlent de configuration de *Workflows*, en assemblant des services paramétrés dans un *Workflow*. Ces auteurs parlent de famille de services comme étant l'ensemble de service qui exhibent de la variabilité et qui peuvent être représentés par les langages de modélisation de la variabilité (modèle de *Features* dans ce cas). Ils proposent alors d'organiser ces services comme des éléments d'une ligne de produit, en mettant en évidence la variabilité exhibée par ces services. Aussi, Mohabbati et ses collègues (Mohabbati et *al.*,2011) proposent une approches de configuration de ligne de produit logiciel orienté service (*SOSP-Service-Oriented Software Product Lines*). Particulièrement, ils utilisent une méthode d'optimisation linéaire pour la sélection du service optimal, en fonction des exigences non-fonctionnelles.

La configuration de service s'apparente alors à la configuration de lignes de produit. La configuration de produit consiste à composer des produits individuels à partir des éléments d'une ligne de produits, en intégrant les éléments communs, et en fixant certains choix par

rapport aux éléments variables, tout en prenant en compte les restrictions relatives à la compatibilité de ces éléments. La variabilité est modélisée de façon explicite dans les lignes de produit logiciel (Clements&Northrop,2001) (Niemelä&Immonen,2007) (Rohleder,2009) (Djebbi,2011), afin de permettre la réutilisation entre les lignes de produit (Ommering,2002). Thompson et Heimdahl (Thompson&Heimdahl,2002) observent que les parties communes et variables peuvent être partagées à travers les lignes de produit. Ainsi, il existe plusieurs modèles de variabilité dont celui de Bühne et ses collègues (Bühne et *al.*,2005) qui proposent une modélisation explicite des informations de variabilité à travers les concepts de variantes et de points de variation dans les lignes de produits. Particulièrement, ils soulignent, comme (Lutz,2000), la définition des contraintes de variabilité correspondant aux contraintes de dépendance entre les exigences variables. Ces contraintes de dépendance sont importantes pour la validité de la ligne de produit.

Mahbub, Afandi et leurs collègues (Mahbub&Spanoudakis,2004) (Afandi et *al.*,2006) se concentrent sur le *monitoring*, lequel est une étape primordiale dans le processus de mesure des attributs de qualité dynamiques, lesquels sont liés à la performance qui varient dans le temps, contrairement à des attributs statiques, tels que la sécurité, le prix, *etc*.

Nous pouvons dire que la qualité de service correspond à la capacité d'un service à répondre au mieux aux exigences non-fonctionnelles des utilisateurs finaux. La qualité de service traitée par la communauté SOC est dédié à des développeurs et concepteurs de services, lesquels étudient plusieurs aspects techniques correspondant aux diverses techniques d'implémentation de la qualité et de son observation. L'expression de la qualité de service par des informations techniques est souvent incompréhensible par les utilisateurs finaux. Une telle qualité ne mentionne pas en quoi permet-elle de satisfaire les exigences non-fonctionnelles des clients. Ces derniers expriment leurs besoins en termes de buts à atteindre, mais non en des termes techniques.

2.2.4.2 *LA QUALITE DE SERVICE DANS L'INGENIERIE DES EXIGENCES*

Le concept de qualité de service est étroitement lié aux exigences non-fonctionnelles (Glinz,2005)(Glinz,2007). Dans la communauté de l'ingénierie des exigences, et particulièrement lorsqu'une approche orientée service est adoptée, la qualité de service est alors modélisé comme les exigences non-fonctionnelles des clients (*NFR-Non Functional Requirements*) (Aiello&Giorgini,2004) (Penserini et *al.*,2006a)(Ma et *al.*,2009).

2.2.4.2.1 DEFINITION DES EXIGENCES NON-FONCTIONNELLES

Plusieurs définitions existent dans la littérature concernant les exigences non-fonctionnelles (*NFR-Non Functional Requirements*). Nous en présentons quelques unes, ci-dessous.

Boehm et ses collègues (Boehm et *al.*,1976), repris plus tard par Keller et ses collègues (Keller et *al.*,1990), définissent les NFR comme des *attributs de qualité*. Robertson et Robertson (Robertson&Robertson,1999) définissent les NFR comme les *propriétés* permettant de rendre le produit plus attractif, plus facile à utiliser, plus rapide ou encore plus fiable. Ils précisent aussi que ces propriétés ne sont pas nécessaires, car elles ne représentent pas les activités principales du produit, mais elles sont recommandées afin de préciser la manière dont les activités principales du produit doivent se réaliser. Selon Thayer, Dorfman et Chung (Thayer&Dorfman,1990)(Chung,1991) les NFR sont les exigences logicielles qui ne décrivent pas ce que le système fait, mais comment elle doit le faire, telles que les exigences de performance. Plus tard, Chung et ses collègues (Chung et *al.*,2000) renomment les NFRs par des buts qualité (*i.e. soft goals*). Après avoir discuté les différentes définitions du NFR, Glinz (Glinz,2007) conclut que cette dernière est un attribut ou une contrainte sur le système.

Plusieurs chercheurs proposent de classer les exigences non-fonctionnelles. Robertson et Robertson (Robertson&Robertson,1999) proposent une classification des NFRs en huit catégories, reprise plus tard par (Cysneiros et *al.*,2001), à savoir : ergonomie, utilisabilité, performance, exigences opérationnelles, portabilité, sécurité, *etc*. Chung et ses collègues (Chung et *al.*,1996), quant à eux, proposent un catalogue de quelques terminologies de NFR. Le travail de Chung, Mylopoulos et Nixon (Mylopoulos et *al.*,1992)(Chung et *al.*,1996) concernant le cadre d'application NFR (*NFR-Framework*) sera expliqué dans la Section 2.2.5.

Nous remarquons qu'il n'existe pas de consensus sur la définition des exigences non-fonctionnelles. Aussi, il n'existe pas de classification universelle ou unique des exigences non-fonctionnelles.

Deux principales approches s'intéressent au traitement des exigences non-fonctionnelles : les approches orientées produit et les approches orientées processus (Mylopoulos et *al.*,1992). Les approches orientées produit, telles que Boehm, Keller, Basili, Hall et leurs collègues (Boehm et *al.*,1978)(Keller et *al.*,1990) (Basili&Musa,1991)(Hall&Fenton,1997) proposent d'évaluer successivement les différents aspects non-fonctionnels (*i.e.* attributs qualité) afin d'affecter au produit un niveau global de qualité. Ces approches incluent des métriques de qualité, afin d'évaluer quantitativement la qualité du système logiciel, en mesurant en quoi le système répond aux exigences non-fonctionnelles. Par exemple, les métriques « nombre de transaction/seconde » et « temps de réponse » sont associées à l'attribut « vitesse ».

Mylopoulos, Chung et leurs collègues (Mylopoulos et *al.*,1992) (Chung et *al.*,2000) proposent de traiter les exigences non-fonctionnelles en parallèle avec les exigences fonctionnelles durant le processus de développement du produit. En effet, différentes décisions de conception peuvent affecter positivement ou négativement certaines exigences non-fonctionnelles. Cette dépendance positive et négative peut d'une part argumenter le fait qu'un produit réponde à certaines exigences non-fonctionnelles, et d'autre part expliquer pourquoi il ne répond pas à ces exigences. Les travaux de ces auteurs seront détaillés dans la Section 2.2.5 (*NFR-framework*).

En résumé, nous pouvons dire que dans le monde des services, les approches de la communauté SOC (*e.g.* (Zeng et *al.*,2003) (Dobson et *al.*,2005) (Mohabbati et *al.*,2011)) sont orientées produit, alors que celles de la communauté RE (*e.g.* (Penserini et *al.*,2006a)(Ma et *al.*,2009)(AitAliSlimane et *al.*,2011)) sont plutôt orientées processus.

2.2.4.2.2 CARACTERISTIQUES DES EXIGENCES NON-FONCTIONNELLES

Selon Chung, Maximilien, Singh et leurs collègues (Chung et *al.*,2000) (Maximilien&Singh,2002), les exigences non-fonctionnelles possèdent trois principales caractéristiques, à savoir :

- les exigences non-fonctionnelles sont subjectives, car elles sont vues, interprétées et évaluées différemment d'une personne à une autre ;
- elles sont également relatives, car leur interprétation et leur importance dépendent fortement du contexte du système considéré. D'après Maximilien et Singh (Maximilien&Singh,2002), la qualité de service dépend fortement de l'importance que l'utilisateur peut donner à un besoin non fonctionnel plutôt qu'à un autre. Ainsi, un service peut avoir plusieurs niveaux de qualité, lesquels dépendent d'une personne à une autre ; et enfin
- les exigences non-fonctionnelles sont interdépendantes, car l'implémentation d'une exigence non-fonctionnelle peut influencer négativement une autre exigence non-fonctionnelle. Par exemple, nous considérons les exigences non-fonctionnelles « vitesse » et « sécurité ». Plus le système est sécurisé, moins il est rapide, car celui-ci prend plus de temps à faire des vérifications.

2.2.5 LE CADRE DE REFERENCE NFR

Dans le développement logiciel, les exigences non-fonctionnelles représentent l'approche pragmatique permettant de construire la qualité d'un système logiciel. Néanmoins, les exigences non-fonctionnelles sont difficiles à adresser, car elles sont interdépendantes, elles

possèdent un large impact sur les fonctionnalités du système, et les exigences non-fonctionnelles peuvent être subjectives.

Ainsi, le cadre de référence NFR (*NFR-framework*) (Mylopoulos et *al.*,1992) (Chung et *al.*,2000) permet de traiter, durant le processus de développement des logiciels, les exigences non-fonctionnelles. Vu leurs caractéristiques (*cf.* Section 2.2.4.2.2), les exigences non-fonctionnelles sont traitées comme des buts potentiellement conflictuels et synergiques. Elles sont représentées sous forme de *buts qualité* (dit aussi *soft goal*) dont les relations d'interdépendance sont capturées dans un graphe. L'impact des décisions est propagé qualitativement à travers le graphe pour déterminer comment le système peut satisfaire au mieux ses exigences non-fonctionnelles. Avant d'expliquer les concepts du cadre NFR, nous proposons de présenter quelques définitions du but.

2.2.5.1 *DEFINITIONS DU BUT*

Le cadre NFR se base sur le concept de but et de son affinement. Avant d'expliquer les concepts de ce cadre, nous proposons de définir d'abord le but, sa taxonomie et son affinement en sous-buts.

2.2.5.1.1 LE BUT

Les exigences provenant des utilisateurs et autres parties prenantes du système sont souvent formulées sous forme de buts désirables. Un but exprime une intention, un objectif que l'on souhaite atteindre et que le système doit réaliser. La communauté d'intelligence artificielle (Nilsson,1971) (Buisson et *al.*,1987) est la première à avoir utilisé le concept de but pour faire du raisonnement sur différentes alternatives visant à atteindre les buts. La communauté de l'ingénierie des exigences (Yue,1987)(Checkland&Holwell,1998) a montré l'utilité de modéliser les exigences par des buts. La gestion des exigences par les buts offre divers avantages dont la stabilité, l'exhaustivité et la technique d'affinement (Van Lamsweerde,2001). Van Lamsweerde (Van Lamsweerde,2000) définit le but comme un objectif que le système doit atteindre à travers une coopération d'agents dans le futur logiciel et dans l'environnement. Il (Van Lamsweerde,2001) différencie le concept de but de celui de l'exigence en précisant que le but est une assertion prescriptive que doit satisfaire le système considéré, tandis que l'exigence est une assertion prescriptive que doit satisfaire la partie logicielle du système uniquement. Anton et ses collègues (Anton et *al.*,1994) affirment que les buts sont les objectifs de haut niveau du métier, de l'organisation ou du système.

Les buts peuvent s'exprimer à plusieurs niveaux d'abstraction : stratégique, tactique, opérationnalisable (Ralyté,2001) et permettent ainsi de capturer, de manière homogène

(expressions de la même nature), des intentions de divers acteurs ayant des enjeux différents. Les buts stratégiques adressent des intentions du monde stratégique de l'entreprise, ils ont une portée globale alors que les buts tactiques sont plus 'locaux' et spécifiques. Les buts opérationnalisables sont ceux qui peuvent être associés sans difficulté à une séquence d'actions, laquelle permet leurs réalisations. Par exemple, « Augmenter le nombre de contrats de 10% », « Effectuer une transaction bancaire », et « Authentifier le client » sont des buts, respectivement, stratégique, tactique et opérationnalisable.

2.2.5.1.2 LA TAXONOMIE DE BUT

Un but peut être un but fonctionnel ou non fonctionnel. Le but fonctionnel définit les services que le système doit fournir. Le but non fonctionnel représente les qualités, à savoir les exigences non-fonctionnelles, telles que la sécurité, la performance et la précision que le système doit satisfaire lorsqu'il fournit ses services (Keller et *al.*,1990). Un but est classé comme un *hard goal* ou un *soft goal*. Un but *hard goal* est celui dont les critères de réalisation sont clairement définis. La satisfaction du but *hard goal* est objective, car elle peut être établie en utilisant des techniques de vérification (Dardenne et *al.*,1993) (Darimon&Van Lamsweerde,1996). Un but *soft goal*, dit aussi *but qualité*[3] par Yu et ses collègues (Yu et *al.*,1995), est celui dont la satisfaction ne peut être établie de façon claire (Mylopoulos et *al.*,1992). Le but *soft goal* est dit « *satisficeable* », par opposition au but *hard goal* qui est dit « *achievable* » (Mylopoulos et *al.*,1992)(Chung et *al.*,2000)(Letier&Van Lamsweerde,2004). Raisonner alors sur le but *soft goal* consiste à se baser sur le concept de *satisficing* (Simon,1981), lequel réside dans la définition d'un ensemble de seuils de satisfaction et d'accepter tout aboutissement au-delà de ces seuils. Jureta et ses collègues (Jureta et *al.*,2006) proposent la taxonomie suivante : (i) le but fonctionnel de type *hard goal* est celui qui décrit ce que le système doit faire ; (ii) le but non fonctionnel de type *hard goal* est celui qui décrit les critères vérifiables sur comment le système devrait fonctionner ; (iii) le but fonctionnel de type *soft goal* est celui qui décrit une exigence subjective du partenaire sur ce que le système doit faire ; et (iv) le but non fonctionnel de type *soft goal* est celui qui décrit de manière subjective et non vérifiable comment le système devrait fonctionner.

Dans cette thèse, nous utiliserons le terme de *but qualité* (dit *soft goal* dans le cadre NFR) pour désigner les exigences non-fonctionnelles, et le terme de *but* pour représenter les *hard goals* qui contribuent à satisfaire partiellement les buts qualité (appelé *satisficing goal* ou *méthode* dans le cadre NFR).

[3] Nous utiliserons le terme de but qualité dans ce rapport de thèse

2.2.5.1.3 L'AFFINEMENT DU BUT

Directement emprunté des méthodes de réduction de problème de l'intelligence artificielle (Nilsson,1971), les graphes *Et*/*Ou* peuvent être utilisés pour capturer les liens d'affinement de buts (Dardenne et *al.*,1991) (Dardenne et *al.*,1993). Le processus d'affinement d'un but en sous-buts est poursuivi jusqu'à atteindre des sous-buts pouvant être satisfaits par une fonctionnalité du système. Les nœuds du graphe représentent les buts et les arcs sont les liens d'affinement.

La représentation graphique des liens d'affinement de buts est illustrée à la Figure 6. Le lien d'affinement *Et* (*cf.* Figure 6-(a)), reliant un but *B1* à un ensemble de sous-buts *B1.1*, *B1.2*, *B1.3*, signifie que la satisfaction de tous les sous-buts *B1.1*, *B1.2* et *B1.3* est suffisante pour satisfaire le but parent *B1*. Le lien d'affinement *Ou* (*cf.* Figure 6-(b)), reliant un but *B2* à un ensemble de sous-buts *B2.1* et *B2.2*, signifie que la satisfaction de l'un des sous-but *B2.1* ou *B2.2* est suffisante pour satisfaire le but parent *B2*. L'affinement *Et* permet d'affiner un but de niveau d'abstraction i en un but de niveau d'abstraction inférieur, i+1, tandis que l'affinement *Ou* introduit la variabilité dans la manière de réaliser un but, en proposant des alternatives de réalisation de but. La combinaison des deux opérateurs aboutit à la construction d'arbres d'affinement *Et*/*Ou* permettant de passer des buts stratégiques à des buts opérationnalisable.

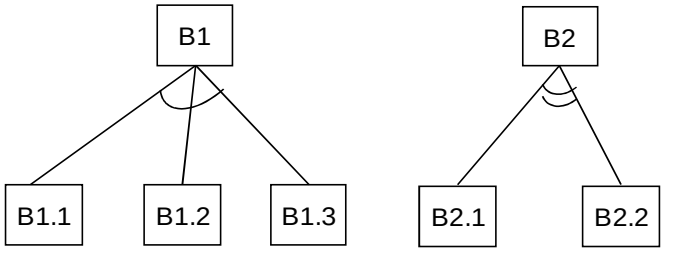

Figure 6. Représentation de l'affinement *Et*/*Ou* des buts

2.2.5.2 *DEFINITIONS DES CONCEPTS DU CADRE NFR*

Le cadre NFR permet de représenter les exigences non-fonctionnelles par des buts qualité. Ces derniers peuvent être affinés et ils peuvent également impacter positivement ou négativement les fonctionnalités du système. Nous présentons dans ce qui suit les différents concepts du cadre NFR.

2.2.5.2.1 LE BUT QUALITE

Dans le cadre NFR (Mylopoulos et *al.*,1992)(Chung et *al.*,2000), le but représente les exigences non-fonctionnelles, telles que la sécurité, la performance et la précision que le

système doit satisfaire. Le but qualité est noté *Type[Paramètre]* par Mylopoulos et ses collègues (Mylopoulos et *al.*,1992)(Chung et *al.*,2000). Le Type correspond aux catégories des exigences non-fonctionnelles existantes, à savoir la sécurité, la performance, la précision, *etc*. Le Paramètre est optionnel et il correspond au sujet (ou à la cible) concerné par le Type. Chaque Type peut être composé de 0 ou plusieurs paramètres. Chaque type peut être associé à 0 ou plusieurs paramètres. Le but qualité est représenté à la Figure 7 sous forme d'un cercle étiqueté avec le nom du but qualité (*i.e.* Type [Paramètre] de la Figure 7-(a) et Type de la Figure 7-(b)).

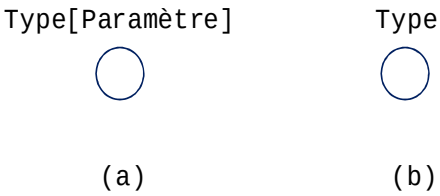

Figure 7. Représentation graphique d'un but qualité dans le cadre NFR.

2.2.5.2.2 L'AFFINEMENT DU BUT QUALITE

Dans le cadre NFR (Mylopoulos et *al.*,1992)(Chung et *al.*,2000), l'affinement du but qualité s'apparente à l'affinement du but présenté ci-dessus (*cf.* Section 2.2.5.1.3). En effet, le but qualité (le type et/ou le paramètre) peut être affiné par des liens *Et/Ou* en d'autres sous-buts qualité dont l'abstraction est plus faible. Néanmoins, cet affinement ne prend pas en compte la satisfaction partielle (*i.e. satisficing*) et la relation d'interdépendance entre les buts qualité. Pour cela, le cadre NFR permet de modéliser la notion de *satisficing* de but qualité, laquelle permet d'exprimer qu'un but qualité parent peut être satisfait dans des limites acceptables, plutôt que de façon absolue.

La notion de *satisficing* est représentée par des *liens de contribution* (représentés par des flèches pleines à la Figure 8) reliant les buts aux buts qualité. En effet, un but peut contribuer *positivement* (« + »), *très positivement* (« ++ »), *négativement* (« - ») ou *très négativement* (« -- ») à la *satisfaction partielle* d'un but qualité. Le Tableau 1 récapitule les liens de contributions.

Tableau 1. Les liens de contribution

Lien de contribution	Signification
« ++ »	un but contribue très positivement à satisfaire un but qualité.
« + »	un but contribue positivement à satisfaire un but qualité.
« -- »	un but contribue très négativement à satisfaire un but qualité.
« - »	un but contribue négativement à satisfaire un but qualité.

2.2.5.2.3 LE LIEN DE CORRELATION

Nous avons vu que les liens de contribution montrent comment les buts peuvent contribuer à la satisfaction partielle des buts qualité. Les liens de corrélation (représentés par des flèches pointillées à la Figure 8), quant à eux, indiquent un effet de bord sur d'autres buts qualité. Ces liens spécifient des contributions positives ou négatives entre les buts et d'autres buts qualité. Particulièrement, lorsque les buts qualité sont interdépendants (*i.e.* conflictuels ou synergiques), un lien de corrélation est précisé entre ces buts qualité. Les liens de corrélation permettent d'aider à l'analyse des compromis.

La criticité du but qualité est indiquée par « ! » pour préciser qu'un but qualité est critique, tandis que le symbole « !! » désigne que le but qualité est très critique.

2.2.5.2.4 LE GRAPHE DES BUTS QUALITE

Le cadre NFR représente l'affinement des buts qualité par un graphe de buts qualité, dit le *Soft goal Interdependency Graph* (*SIG*) (Chung et *al.*,2000). La Figure 8 présente un exemple de SIG (originaire de (Chung&Nixon,1995)) illustrant un exemple relatif à la sécurisation d'un compte bancaire. Celui-ci montre que le but Authentifier[Compte] permet de contribuer positivement à la satisfaction de la confidentialité. Ce lien de contribution est représenté à la Figure 8 par une flèche pleine indiquant que le but qualité Confidentialité[Compte] impacte positivement l'authentification du compte. D'autre part, le but Authentifier[Compte] est en synergie avec la précision du compte, d'où le lien de corrélation positif entre les deux buts, à la Figure 8. Aussi, l'authentification par biométrie (*i.e.* Biométrie[Compte]), qui est une alternative possible pour Authentifier[Compte], est en conflit avec le but qualité Performance[Compte], et plus particulièrement au Temps[Compte]. Ainsi, le lien de corrélation négatif entre Biométrie[Compte] et Temps[Compte] est illustré à la Figure 8. Par conséquent, si le concepteur choisit d'implémenter une authentification par biométrie, il aura certes de la confidentialité et de la précision (« + »), mais il aura moins de performance («-»). Dans le cadre NFR, les buts peuvent également être affinés, comme le montre la Figure 8.

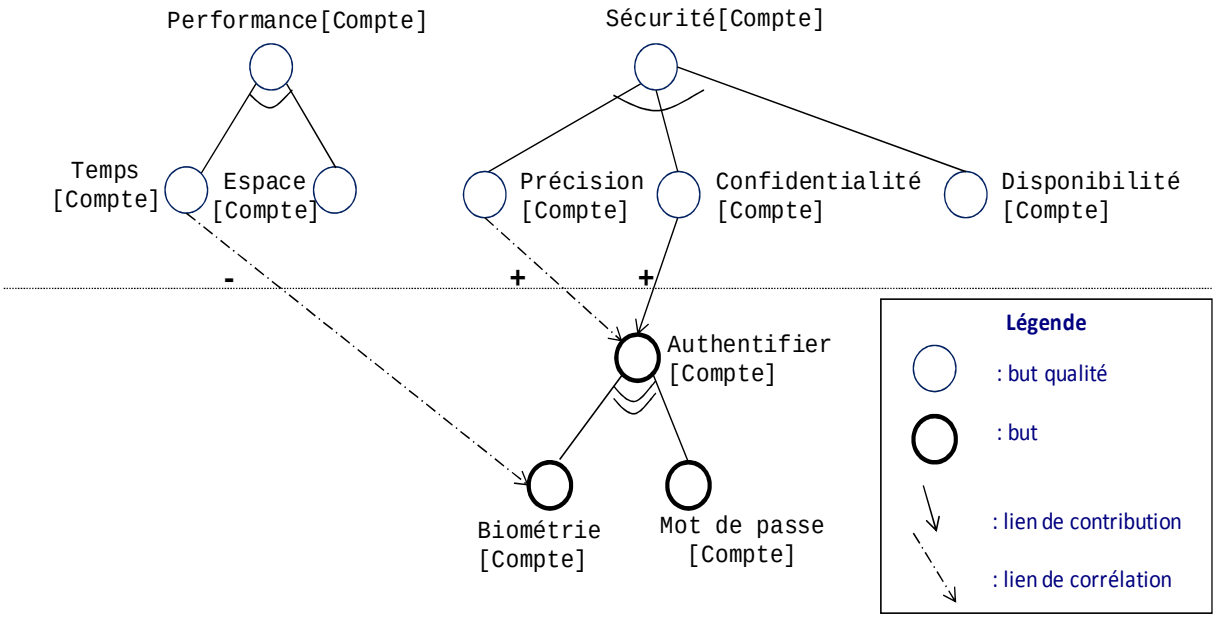

Figure 8. Exemple de SIG

Le cadre NFR permet l'acquisition et la représentation des connaissances concernant les différentes exigences non-fonctionnelles du domaine à développer. Ces connaissances sont collectées et cataloguées sous forme de buts qualité et de buts, qui sont mis à la disposition des développeurs et concepteurs. Le cadre NFR fournit alors un catalogue de quelques exigences non-fonctionnelles, telles que celle présentées à la Figure 9. Ce cadre fournit aussi une méthodologie d'intégration des NFRs dans le processus de développement des logiciels (Chung&Nixon,1995). Le concepteur peut choisir d'utiliser et de réutiliser le catalogue existant, ou de proposer son propre catalogue, tout en précisant les liens de contribution et de corrélation.

La méthodologie d'intégration des exigences non-fonctionnelles (Chung&Nixon,1995) consiste à : (1) identifier les buts qualité et leurs affinements ; (2) identifier les alternatives ou les buts ; (3) identifier les conflits et les synergies (alliance) ; (4) définir les buts qualité critiques ; et (5) évaluer les buts qualité et sélectionner les alternatives.

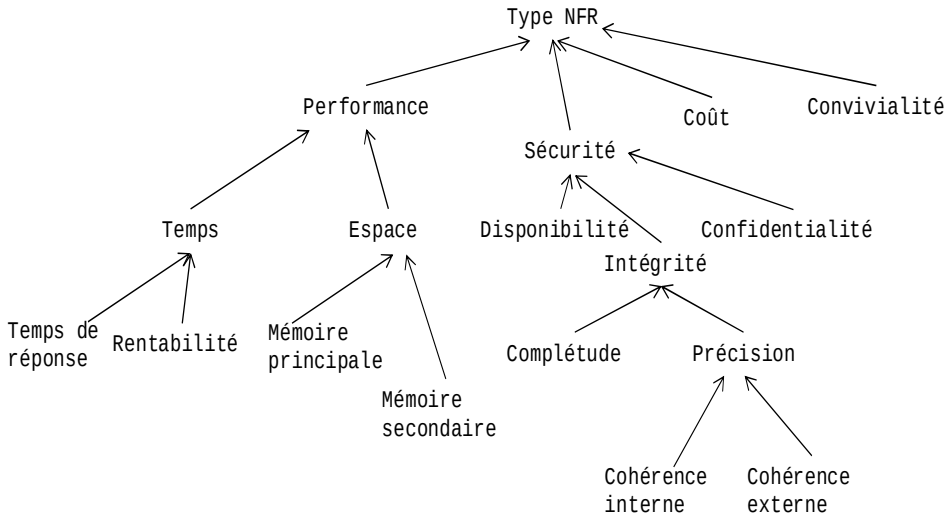

Figure 9. Catalogue de quelques terminologies de NFR (Chung et *al.*,1996)

2.2.5.2.5 EVALUATION DES BUTS QUALITE

L'évaluation des buts qualité est effectuée en utilisant la procédure d'étiquetage (Mylopoulos et *al.*,1992). Celle-ci consiste à étiqueter les buts qualité feuilles du SIG, en se basant sur combien ils sont partiellement satisfaits (*Satisficed*, *Denied*, *Conflicting*, *Undetermined*) et sur leur criticité (« ! » ou « !! »). Il s'agit par la suite de propager, du bas vers le haut, les étiquettes à travers le graphe tout en considérant les différents liens de contribution.

Plusieurs travaux ont adopté le cadre NFR : Supakkul et Chung (Supakkul&Chung,2004) l'ont utilisé pour intégrer la qualité dans les diagrammes des cas d'utilisation et faciliter ainsi l'adoption du cadre NFR dans l'industrie. Rohleder et leurs collègues (Rohleder et *al.*,2009) pour considérer la qualité dans les différente étapes du cycle de vie d'un produit (*PLM-Product Lifecycle Management*).

Dans l'ingénierie des applications à base de service, Aiello, Penserini, Ma et leurs collègues (Aiello&Giorgini,2004) (Penserini et *al.*,2006) (Ma et *al.*,2009) modélisent la qualité de service comme l'ensemble des buts qualité que le service permet de satisfaire partiellement.

Particulièrement, Ma et ses collègues (Ma et *al.*,2009) proposent une ontologie de QoS des services métier en se basant sur le cadre NFR.

Les buts qualité sont considérés comme les critères de sélection des services par les approches de Penserini, Ma et leurs collègues (Penserini et *al.*,2006)(Ma et *al.*,2009). Ces approches proposent de sélectionner, lors de la conception du système logiciel, les services qui d'une part répondent aux exigences fonctionnelles des clients, et d'autre part ont le plus de similitudes avec les exigences non-fonctionnelles des clients (Penserini et *al.*,2006) (Ma et

al.,2009). Ma et ses collègues (Ma et *al.*,2009), par exemple, proposent d'utiliser une méthode multicritères d'aide à la décision (*MCDM-Multicriteria Decision Making*) pour sélectionner les services.

Le cadre NFR (Chung et *al.*,2000) est une excellente ressource pour notre recherche concernant l'impact des exigences non-fonctionnelles sur les services intentionnels (*cf.* Chapitre 6). Néanmoins, le cadre NFR seul ne répond pas aux besoins de consensus qualité que nous recherchons. En effet, notre objectif est d'exprimer les seuils de satisfaction qualitatifs, compréhensibles par l'utilisateur final, par des valeurs de qualité, fournies par les services logiciels. Pour cela, nous proposons d'enrichir le cadre afin qu'il réponde à nos besoins (*cf.* Chapitre 5).

2.3 *CADRE DE REFERENCE POUR LA QUALITE DE SERVICE*

Le cadre de référence multidimensionnel, proposé à la Figure 10, s'interroge sur la notion de qualité de service et permet de comparer les différentes approches traitées dans cette thèse. Ce cadre est inspiré de (Rolland et *al.*,1998a) (Rolland et *al.*,1998b) et il est composé de quatre vues différentes et complémentaires. Chaque vue permet d'analyser un aspect particulier de la qualité de service en se posant une question fondamentale. Les quatre questions posées sont les suivantes :

- Quoi ? correspondant à l'*objet*, c'est à dire les éléments ou les connaissances exprimés par la qualité de service.
- Pourquoi ? correspondant aux *buts* recherchés par l'approche qui traite de la qualité de services.
- Comment ? correspondant à la *méthode* mise en œuvre pour atteindre ces buts.
- Par quel moyen ? correspondant aux *outils* ou techniques utilisés.

Chaque vue est caractérisée et mesurée à l'aide d'un ensemble d'attributs. Ces derniers sont définis à partir de l'analyse effectuée sur quelques approches qui traitent de la qualité de service. Nous avons considéré aussi bien des approches de la communauté SOC que celles de l'ingénierie des exigences. Les attributs possèdent des valeurs définies dans un domaine. Les valeurs peuvent être de type prédéfini (entier, booléen…), de type énuméré (Enum {x, y}) ou de type structuré (Ensemble ()).

Par exemple, la vue objet comprend deux attributs (*cf.* Tableau 2) dont l'entité et la priorisation. Ces attributs sont expliqués en détail dans la suite du document.

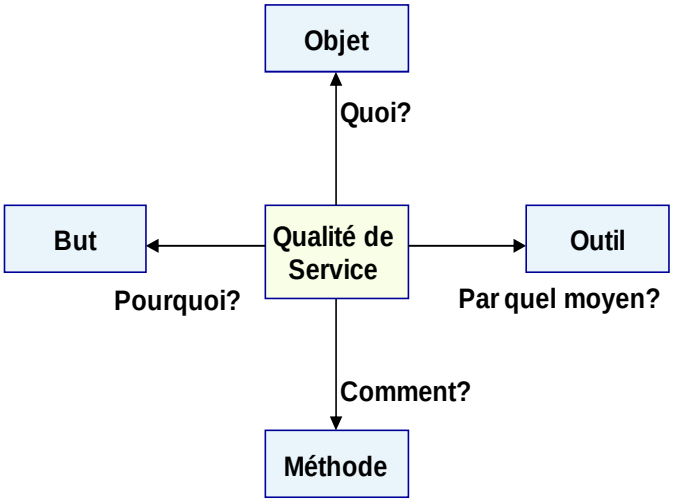

Figure 10. Le cadre de référence de la qualité de service

Tableau 2. Exemple d'attributs, domaines et exemples de valeurs

Attribut	Domaine	Exemple de valeurs
Entité	Entité : Enum {logiciel, applicatif, intentionnel}	intentionnel
Priorisation	Priorisation : Booléen	oui

Chacune des vues proposée par le cadre de référence, à savoir objet, but, méthode et outil, ainsi que leurs attributs, sont décrits dans la suite de cette section.

2.3.1 LA VUE OBJET

La vue *objet* consiste à détailler comment la communauté de l'ingénierie des services et celle des exigences définissent la qualité de service (on parlera plus simplement d'entités), ainsi que les différents paramètres associés à sa définition.

Nous proposons d'associer à la vue objet les attributs suivants : (i) l'entité ; (ii) le consensus ; et (iii) la priorisation. Le reste de la section présente en détail chacun des attributs.

2.3.1.1 *ENTITE*

Nous définissons trois perspectives pour la qualité de service, à savoir : le point de vue logiciel ; le point de vue applicatif ; et le point de vue intentionnel.

Le point de vue logiciel :

Dans le point de vue logiciel, le terme de qualité de service (*QoS- Quality of Service*) se réfère aux propriétés non-fonctionnelles des services Web, telles que la performance, la fiabilité et les exigences reliées au réseau (Ran,2003) (Sumra&Arulazi,2003) (W3C,2003). Ainsi, la qualité de service correspond à des métriques calculées sur les services logiciels

(Maximilien&Singh,2004a) (Zeng et *al.*,2003) (Ran,2003) (Herssens et *al.*,2008) (SeCSE,2009).

Le point de vue applicatif :

Dans le point de vue applicatif, la qualité de service est prise en compte lors de la conception d'application à base de services. Penserini, Aiello et leurs collègues (Penserini et *al.*,2006a) (Aiello&Giorgini,2004) suggèrent de modéliser la qualité de service parallèlement à la modélisation des services. La qualité de service est modélisée comme un but qualité que les services permettent de satisfaire partiellement. La définition de la qualité de service reste donc au niveau applicatif.

Le point de vue intentionnel :

Dans le point de vue intentionnel, la qualité de service est définie au niveau intentionnel. Rohleder (Rohleder,2009) décrit la qualité de service comme l'ensemble des buts qualité qui répondent aux exigences non-fonctionnelles des utilisateurs. Le point de vue intentionnel offre une définition orientée but, compréhensible par les utilisateurs et un mode de calcul de la qualité de service dirigé par les buts.

L'attribut entité peut donc être de trois types différents comme le montre la définition ci-dessous :

Entité : Enum {logiciel, applicatif, intentionnel}

2.3.1.2 *CONSENSUS*

Afin de faciliter le processus de sélection, les fournisseurs et les clients doivent avoir une définition commune des concepts de la qualité de service (Maximilien&Singh,2004a) (Penserini et *al.*,2006a) (Herssens et *al.*,2008) (Zachos et *al.*,2009). Pour cela, la majorité des approches (Ran,2003)(Maximilien&Singh,2004b)(Dobson et *al.*,2005)(Wang et *al.*,2006)(Ma, et *al.*,2009) proposent des ontologies de qualité, afin de partager le même vocabulaire qualité entre les fournisseurs et les clients de services. La qualité de service est considérée par les fournisseurs de service pour décrire la qualité de leurs services et par les clients pour définir leurs exigences non-fonctionnelles.

L'attribut consensus est défini comme une valeur booléenne permettant de préciser si l'approche gère, ou non, le consensus entre les qualités publiées et celles demandées par les clients. Il est résumé dans ce qui suit :

Consensus : Booléen

2.3.1.3 *PRIORISATION*

Les exigences non-fonctionnelles des clients peuvent contenir des qualités qui ne peuvent être satisfaites simultanément. Par exemple, la sécurité et la performance se corrèlent négativement : si un service satisfait un niveau important de sécurité (en implémentant, par exemple, des algorithmes de cryptage) alors la performance de ce service diminue (temps de réponse du service augmente). Par conséquent, les clients doivent préciser leurs préférences par rapport aux qualités voulues.

Zeng, Herssens, Rohleder et leurs collègues (Zeng et *al.*,2003) (Herssens et *al.*,2008) (Rohleder,2009) proposent d'utiliser un système de poids (ou de coefficient (Rohleder,2009)) afin de prioriser leurs qualités. Un poids est une valeur numérique associée à chaque qualité. La qualité la plus favorisée reçoit le poids le plus grand permettant ainsi, lors du processus de sélection, de choisir le service possédant la qualité ayant le plus grand poids.

L'attribut priorisation est défini comme une valeur booléenne permettant de préciser si l'approche gère, ou non, la priorisation des différentes qualités. Il est défini comme suit :

Priorisation : Booléen

2.3.2 LA VUE BUT

La vue *but* consiste à détailler les buts recherchés par l'intégration de la qualité de service dans les approches à base de service et celles de l'ingénierie des exigences. Elle compte un seul attribut intitulé *but* et propose de répondre à la question « pourquoi la qualité de service est elle utilisée ? ».

La qualité de service est modélisée par toutes des approches étudiées dans ce rapport. Par exemple Penserini, Aiello, Rohleder et leurs collègues (Penserini et *al.*,2006a) (Aiello&Giorgini,2004) (Rohleder,2009) modélisent la qualité comme un but qualité à satisfaire partiellement, tandis que Maximilien, Dobson et leurs collègues (Maximilien&Singh,2004a) (Dobson et *al.*,2005) proposent des ontologies de qualité, lesquelles regroupent tous les concepts relatifs à la qualité de service.

La qualité de service peut concerner aussi bien un service atomique qu'un service composite. Pour cela, l'approche de Zeng et ses collègues (Zeng et *al.*,2003) propose de calculer la qualité du service composite. Aiello, Penserini et leurs collègues (Aiello&Giorgini,2004) (Penserini et *al.*,2006a) se sont intéressés à l'évaluation de la satisfaction partielle des buts qualité. Rohleder (Rohleder,2009) propose également de calculer la QoS des variantes.

Plusieurs services peuvent fournir les mêmes fonctionnalités, tout en fournissant des qualités différentes. La majorité des travaux traités dans ce rapport (Zeng et *al.*,2003) (Maximilien&Singh,2004a) (Penserini et *al.*,2006a) (Zachos et *al.*,2009) se concentrent sur la sélection de service, en se basant sur les critères de qualité. Zeng et ses collègues (Zeng et *al.*,2003) font de la sélection dans la perspective de composer dynamiquement les services ayant les meilleures qualités. D'autres (Maximilien&Singh,2004a) (Penserini et *al.*,2006a) (Zachos et *al.*,2009) proposent de faire de la sélection de services en prenant en compte les préférences qualités des utilisateurs.

Il existe néanmoins d'autres travaux qui font de la configuration de services, en se basant sur les critères de qualité. La configuration (Dustdar&Schreiner,2005) consiste à adapter le service composite en fonction des exigences de l'utilisateur final. En effet, le service à variation sélectionné peut être configuré afin d'adapter les variantes du service aux exigences non-fonctionnelles des clients (*i.e.* éliminer les variantes en fonction des critères de qualité). Rohleder (Rohleder,2009) propose, dans ses travaux, de faire la configuration des variantes de lignes de produit.

L'attribut but met en évidence les objectifs recherchés par les différentes approches. Il est récapitulé comme suit :

But : Ensemble {modéliser, évaluer, sélectionner, configurer}

2.3.3 LA VUE METHODE

La vue méthode consiste à définir la démarche permettant de réaliser les buts cités dans la section précédente (*cf.* Section 2.3.2). Elle est composée de plusieurs attributs, à savoir la méthode de modélisation de la qualité ; la méthode d'évaluation de la qualité ; la stratégie d'évaluation ; et la méthode de sélection ou de configuration. Chacun de ces attributs est détaillé dans cette section.

2.3.3.1 *METHODE DE MODELISATION*

Diverses façons de modéliser la qualité de service sont proposées par les approches traitées dans ce rapport.

Certaines approches proposent de modéliser de façon formelle la qualité via le concept de but qualité (*i.e. soft goal*) (Aiello&Giorgini,2004) (Penserini et *al.*,2006a) (Rohleder,2009), ou d'ontologie (Maximilien&Singh,2004a) (Dobson et *al.*,2005) (Zachos et *al.*,2009). D'autres approches (Herssens et *al.*,2008) (Zeng et *al.*,2003) suggèrent de modéliser la qualité soit de façon semi-formelle en proposant un modèle UML (Herssens et *al.*,2008), soit de

façon informelle via un vecteur qualité (Zeng et *al.*,2003). Ce dernier est une représentation multidimensionnelle de quelques critères de qualité.

L'attribut méthode de modélisation précise le type de modélisation utilisé par les diverses approches. Il est résumé ci-dessous :

Méthode de modélisation : Enum {formelle, semi-formelle, informelle}

2.3.3.2 *METHODE ET STRATEGIE D'EVALUATION*

La sélection d'un service, en se basant sur les critères de qualité, consiste d'abord à évaluer la qualité de ce service. Deux méthodes d'évaluation sont utilisées par les différentes approches, classées selon leur nature : quantitative ou qualitative.

L'évaluation quantitative proposée par (Zeng et *al.*,2003) (Maximilien&Singh,2004a) (Zachos et *al.*,2009) (Herssens et *al.*,2008) est associé à des mesures quantitatives (des valeurs numériques) correspondant aux qualités.

L'évaluation qualitative, proposée par (Aiello&Giorgini,2004) (Penserini et *al.*,2006a) (Rohleder,2009), consiste à évaluer la satisfaction partielle des buts qualité correspondant à la qualité de service. Cette évaluation repose sur l'interprétation, le jugement et la connaissance des concepteurs et/ou des clients du service. Ce sont des appréciations subjectives de ces acteurs qui permettent de conclure à la satisfaction ou non de la qualité. Dans (Aiello&Giorgini,2004), les valeurs sont définies de façon plus précise, ce qui réduit le degré de subjectivité. Cependant, la valeur retenue dépend néanmoins de l'appréciation personnelle des experts impliqués.

L'évaluation qualitative ou quantitative de la qualité de service peut être locale ou globale. L'évaluation locale se focalise principalement sur l'estimation de la qualité du service atomique, ou celle du service composite. Pour ce dernier, l'évaluation est locale, car les approches (Penserini et *al.*,2006a) (Aiello&Giorgini,2004) (Rohleder,2009) ne considèrent pas le cas de contribution de plusieurs services, constituant le service composite, à la même qualité.

L'évaluation globale, proposée notamment par (Zeng et *al.*,2003), s'intéresse à calculer la qualité du service composite, en fonction de celle des services atomiques. Cette approche propose des formules de calcul permettant, à partir de la qualité des services atomiques, de calculer la qualité du service composite.

L'attribut méthode et stratégie d'évaluation est défini comme suit :

Méthode d'évaluation : Ensemble {qualitative, quantitative}

Stratégie d'évaluation : Ensemble {locale, globale}

2.3.3.3 *METHODE DE SELECTION/CONFIGURATION*

La sélection ou la configuration d'un service, en se basant sur les critères de qualité, consiste à appliquer des technique permettant de choisir, respectivement, parmi les services et les configurations possédant les meilleures qualités ou ceux répondant le mieux aux préférences qualité des clients.

Herssens, Zeng et leurs collègues (Herssens et *al.*,2008) (Zeng et *al.*,2003) utilisent les méthodes multicritères d'aide à la décision (*MCDA-Multi Criteria Decision Analysis*) afin de classer les services selon leurs qualités. Ces derniers (Zeng et *al.*,2003) utilisent également la programmation linéaire (PL) pour traiter le cas d'une optimisation globale, correspondant à la génération, dans un diagramme d'états, de plusieurs plans d'exécution pour divers chemins d'exécution. D'autres approches (Maximilien&Singh,2004a) (Zachos et *al.*,2009) utilisent des algorithmes de similarité (*i.e. matching algorithms*), afin de sélectionner les services dont les qualités possèdent une correspondance sémantique avec les exigences non-fonctionnelles du client. Rohleder (Rohleder,2009) propose d'utiliser des mesures de similarité afin de définir la configuration du produit selon le profil qualité du produit souhaité.

L'attribut méthode de sélection permet de préciser la méthode utilisée par les approches pour sélectionner les services selon leurs qualités. Il est résumé ci-dessous :

Méthode de sélection : Ensemble {MCDA, PL, algorithme de similarité}

2.3.3.4 *CONTRAINTE DE DEPENDANCE*

Dans le cadre de composition de services incluant plusieurs variations métier, la configuration, basée sur les critères de qualité, consiste à supprimer de la composition les variantes qui ne permettent pas de satisfaire au mieux les critères de qualité. Néanmoins, comme le montre plusieurs travaux de recherche de configuration de lignes de produit (Bühne et *al.*, 2005)(Djebbi,2011), des contraintes de dépendance entre variantes sont à prendre en compte durant le processus de configuration.

Ben Hamida et ses collègues (Ben Hamida et *al.*,2008) parlent de dépendance qui renseigne sur la nécessité de charger des services et une dépendance représentant la possibilité de choisir un service ou un autre. Verma et ses collègues (Verma et *al.*, 2004) considèrent que dans un flux de processus, la sélection de services Web implique la prise en compte des dépendances définies préalablement dans les services sélectionnés. Dans les lignes de produits logiciels, deux type de contraintes de dépendance sont considérées pour gérer la variabilité des exigences fonctionnelles, à savoir la contrainte de dépendance d'inclusion et celle d'exclusion (Bühne et *al.*, 2005) (Djebbi,2011). La modélisation de la variabilité permet de

répondre à un souci de réutilisabilité et d'adaptation à un contexte particulier. Néanmoins, elle implique l'existence de contraintes de dépendance entre les exigences variables.

L'attribut contrainte de dépendance est défini comme une valeur booléenne permettant de préciser si l'approche gère, ou non, les contraintes de dépendance. Il est définit comme suit :

Contrainte de dépendance : Booléen

2.3.4 LA VUE OUTIL

La vue outil indique les techniques utilisées par les approches pour modéliser la qualité ou sélectionner les services. Elle comporte deux attributs, à savoir l'outil de modélisation et l'outil de sélection.

2.3.4.1 *OUTIL DE MODELISATION*

La modélisation de la qualité de service, proposée par les diverses approches étudiées dans ce rapport se basent sur divers outils de modélisation.

Herssens et ses collègues (Herssens et *al.*,2008) utilisent le langage UML pour modéliser la qualité de service, tandis que Zeng et ses collègues (Zeng et *al.*,2003) expriment la qualité de service par un vecteur qualité composé des critères de qualité considérés.

Maximilien, Zachos et leurs collègues (Maximilien&Singh,2004a) (Zachos et *al.*,2009) proposent de modéliser la qualité sous forme d'ontologies : les premiers se basent sur XML et les derniers utilisent OWL-S.

Aiello, Penserini, Rohleder et leurs collègues (Aiello&Giorgini,2004) (Penserini et *al.*,2006a) (Rohleder,2009) proposent de représenter la qualité de service en utilisant un langage orienté but : (Aiello&Giorgini,2004) (Penserini et *al.*,2006a) utilisent TROPOS, tandis que (Rohleder,2009) exploite le cadre NFR.

L'attribut modèle de représentation indique l'outil utilisé pour modéliser la qualité. Il est défini comme suit :

Outil de modélisation : Enum {UML, vecteur, ontologie, langage orienté but}

2.3.4.2 *OUTIL DE SELECTION*

Herssens et ses collègues (Herssens et *al.*,2008) se basent sur la technique *Fuzzy MCDA*, tandis que Zeng et ses collègues (Zeng et *al.*,2003) utilisent la technique SAW (*Simple Additive Weighting*) et la programmation linéaire (PL).

Maximilien et Singh (Maximilien&Singh,2004a) utilisent le langage *Python* pour implémenter l'algorithme de sélection de services. Zachos et ses collègues (Zachos et *al.*,2009) utilisent les outils UCaRE et EDDiE pour effectuer la sélection des services.

L'attribut outil de sélection permet de préciser la technique, l'outil ou le langage utilisé par les approches pour sélectionner les services, selon leurs qualités. Il est résumé ci-dessous :

Outil de sélection : Ensemble {outil (valeur), langage (valeur), technique (valeur)}

2.3.5 SYNTHESE DES QUATRE VUES

Le cadre de référence, illustré à la Figure 10, synthétise les principaux attributs, déduits de l'analyse effectuée sur quelques approches de l'ingénierie des services, lesquelles traitent de la qualité de service. Ces attributs sont organisés suivant quatre vues interdépendantes : objet, but, méthode et outil. Ceci permet d'organiser le classement de six approches de l'ingénierie des services, lesquelles traitent de la qualité de service. Ces vues ainsi que leurs attributs sont récapitulés dans le Cadre 1.

Vue Objet

Entité : Enum {logiciel, applicatif, intentionnel}

Consensus : Booléen

Priorisation : Booléen

Vue But

But : Ensemble {modéliser, évaluer, sélectionner, configurer}

Vue Méthode

Méthode de modélisation : Enum {formelle, semi-formelle, informelle}

Méthode d'évaluation: Ensemble {qualitative, quantitative}

Stratégie d'évaluation: Ensemble {locale, globale}

Méthode de sélection/configuration : Ensemble {algorithme de similarité, Ensemble {MCDA, PL, algorithme de correspondance}}

Contrainte de dépendance : booléen

Vue Outil

Outil de modélisation : Enum {UML, vecteur, ontologie, langage orienté but}

Outil de sélection : Ensemble {outil (valeur), langage (valeur), technique (valeur)}

Cadre 1. Résumé du cadre de référence

2.4 *POSITIONNEMENT DE SEPT APPROCHES EN FONCTION DU CADRE DE REFERENCE*

Cette section présente une évaluation de sept approches, lesquelles traitent de la qualité de service. Nous nous sommes intéressés aussi bien aux approches appartenant à la communauté SOC, qu'à celles de l'ingénierie des exigences. Ces approches sont sélectionnées d'une part, pour leur intérêt particulier concernant la prise en compte de la qualité de services dans la conception de système logiciel à forte variabilité, et d'autre part parce qu'elles forment, selon nous, un ensemble représentatif de l'état de l'art, correspondant à la problématique présentée dans le premier chapitre.

Cette partie décrit brièvement ces approches et les positionne par rapport au cadre de référence décrit ci-dessus (*cf.* Figure 10). Les sept approches sont les suivantes :

1. L'approche du projet SeCSE (Zachos et *al.*,2009);
2. L'approche de Maximilien et Singh (Maximilien&Singh,2004a);
3. L'approche de Zeng et ses collègues (Zeng et *al.*,2003);
4. L'approche d'Aiello et Giorgini (Aiello&Giorgini,2004);
5. L'approche de Penserini et ses collègues (Penserini et *al.*,2006a);
6. L'approche de Herssens et ses collègues (Herssens et *al.*,2008) ;
7. L'approche de Rohleder dans les lignes de produits (Rohleder,2009).

2.4.1 L'APPROCHE DU PROJET SECSE

Le projet SeCSE (*Service-Centric System Engineering*) (SeCSE,2009) propose des méthodes, des outils et des techniques aux intégrateurs de systèmes et aux fournisseurs de services, permettant de supporter le développement de systèmes à base de services.

Particulièrement, SeCSE supporte un processus itératif de découverte des exigences. En effet, l'analyste construit des requêtes de services, à partir des spécifications des exigences. Ces requêtes permettent de découvrir les services correspondant à ces exigences. Les descriptions des services trouvés sont utilisées par les analystes pour affiner et compléter la spécification initiale des exigences et effectuer par la suite une découverte plus précise des services. Cette approche se base sur les observations de Fischer et ses collègues (Fisher et *al.*,1991), lesquels considèrent que la conception des requêtes est améliorée de façon incrémentale grâce aux critiques des résultats obtenus des anciennes requêtes. Ce retour d'information pertinent (*relevance feedback*) permet aux analystes et aux clients de spécifier de nouvelles exigences et de reformuler les exigences existantes afin d'augmenter la probabilité de sélection des services adéquats.

Dans leurs anciens travaux Zachos et ses collègues (Zachos et *al.*,2007) (Zachos et *al.*,2008a) utilisent ce processus itératif pour compléter la spécification des exigences fonctionnelles. Vu l'importance de la qualité de service dans les systèmes à base de services, le projet SeCSE propose de prendre en compte les exigences non-fonctionnelles correspondant à la qualité de service (Zachos et *al.*,2008b). Pour supporter ce processus itératif, l'équipe de SeCSE a développé un environnement composé de plusieurs parties :

1. *L'annuaire des services.* Le projet SeCSE propose un mécanisme extensible de spécification des services qui se base sur les facettes (Walkerdine et *al.*,2007). Un service est décrit selon plusieurs facettes dont la signature, la description et la facette

QoS décrivant les caractéristiques non-fonctionnelles du service. Le fournisseur de services est responsable de fournir ces facettes en utilisant l'outil de spécification de services (Sawyer et *al.*,2005). Cependant, la facette QoS peut être fournie par une tierce partie qui s'occupe d'observer (*monitoring*) les services et de fournir des données sur leurs qualités. La facette QoS permet d'affiner la sélection des services qui ont déjà été découverts grâce à la facette description (Zachos et *al.*,2008b). Dobson et ses collègues (Dobson et *al.*,2005) proposent aux fournisseurs de services de partager une ontologie, dite *QoSOnt*, permettant de décrire les caractéristiques non-fonctionnelles du service.
QoSOnt est partagé en trois couches : la première couche décrit les concepts de base des QoS. Cette dernière est définie comme un attribut mesurable (temps de réponse) ou non mesurable (adhérence à un protocole). Les attributs mesurables possèdent une ou plusieurs métriques associées. Les attributs ont des valeurs pouvant être exprimées dans des unités spécifiques. Les règles de conversion entre unités, jugées nécessaires, sont aussi définies. La deuxième couche s'intéresse à mettre en évidence la hiérarchie entre les divers attributs (par exemple, sécurité, performance, *etc*.), ainsi que leurs métriques respectives. La troisième couche précise le domaine auquel les attributs QoS sont reliés (domaine des réseaux, domaine des services, *etc*.).

2. *La spécification des exigences*. Le projet SeCSE propose un outil de spécification des exigences et des requêtes de service nommé *UCaRE*. L'outil UCaRE permet aux analystes de spécifier leurs exigences et de créer des requêtes de services, à partir de ces spécifications (Zachos et *al.*,2006). La spécification de la qualité de service se fait de deux manières complémentaires (Zachos et *al.*,2008b) : la première en spécifiant en langage naturel les exigences non-fonctionnelles, la deuxième en précisant les critères mesurables (*MFC- Measurable Fit Criteria*) correspondants à ces exigences, ainsi que les contraintes sur les valeurs d'acceptation (minimale et maximale). La spécification des MFC est alignée aux attributs mesurables définis dans QoSOnt (Dobson et *al.*,2005). L'analyste sélectionne à partir de la spécification des exigences les informations qu'il souhaite inclure dans la requête de service.

3. *La découverte de services.* Le projet SeCSE (SeCSE,2009) propose un algorithme de découverte de services appelé *EDDiE*. L'outil EDDiE utilise des techniques permettant de découvrir les services à partir des requêtes de services. La facette QoS, associée aux services, est comparée aux MFC correspondant aux exigences non-fonctionnelles spécifiées par les analystes. L'utilisation de cette facette permet d'affiner la sélection des services qui ont déjà été découverts grâce à la facette

description. L'outil EDDiE se base sur le système de références lexicales croisées WordNet (WordNet,2009). Ces techniques consistent (i) à préciser la sémantique de chaque terme de la requête ; (ii) à rajouter des termes (ayant le même sens ou ayant un sens similaire) à la requête de service afin de la rendre plus complète ; et (iii) à comparer tous les termes de la requête expansée aux termes des services.

Les services découverts sont présentés aux analystes et aux clients des services en utilisant l'explorateur des services (*Service Explorer Component*). Celui-ci affiche, de façon ordonnée, les services qui correspondent aux valeurs d'acceptation exigées.

En résumé, l'approche du projet SeCSE (Zachos et *al.*,2008b) propose un outil de découverte de services en se basant sur les exigences des concepteurs (ou clients). SeCSE modélise la qualité de service au niveau fournisseur comme un attribut qualité (mesurable et non mesurable) suivant l'ontologie QoSOnt. Il modélise également la qualité au niveau analyste (ou client) comme étant une exigence non-fonctionnelle spécifiée en langage naturel et comme un attribut qualité mesurable aligné à l'ontologie QoSOnt. Le projet SeCSE supporte des algorithmes de similarité se basant sur WordNet et QoSOnt pour augmenter la probabilité de découverte des services.

Le positionnement de l'approche du projet SeCSE, par rapport au cadre de référence proposé plus haut, est présenté dans le Cadre 2.

Vue Objet
Entité : applicatif
Consensus : Oui
Priorisation : Non
Vue But
But : modéliser, sélectionner
Vue Méthode
Méthode de modélisation : formelle
Méthode d'évaluation : -
Stratégie d'évaluation : -
Méthode de sélection: algorithme de similarité
Contrainte de dépendance : non
Vue Outil
Outil de modélisation : ontologie
Outil de sélection : outil (UCaRE,EDDiE)

Cadre 2. Positionnement de l'approche du projet SeCSE

2.4.2 L'APPROCHE DE MAXIMILIEN ET SINGH

Maximilien et Singh (Maximilien&Singh,2004a) proposent une approche permettant de traiter la sélection des services durant l'exécution du service composite. La sélection d'un

service se base sur les attributs non fonctionnels. Le service peut être remplacé par un autre, à l'exécution, s'il ne répond pas aux préférences des clients.

Le processus de sélection proposé est composé de deux parties principales, à savoir : (1) une ontologie de qualité de services ; (2) une politique QoS ; et (3) un algorithme de sélection.

1. *Une ontologie QoS.* Les auteurs décrivent la QoS à travers une ontologie qui abstrait ses caractéristiques. Les politiques QoS des fournisseurs et les préférences des clients sont exprimées en utilisant les concepts de l'ontologie. Celle-ci permet d'effectuer une correspondance sémantique entre les politiques QoS publiées par les fournisseurs et les préférences QoS des clients. L'ontologie est décrite dans (Maximilien&Singh,2004b) et regroupe l'ensemble des concepts associés à la qualité, tels que *Quality, QMeasurement* et *QRelationship*. La qualité de service (*Quality*) est définie par ces auteurs comme étant une caractéristique non-fonctionnelle mesurable d'un service. Des indications sur comment mesurer la qualité sont précisées (*QMeasurement*) : la mesure peut être objective (par un calcul automatique) ou subjective (par une intervention humaine). La mesure a une période de validité. Les relations entre qualités sont définies afin de préciser les dépendances entre elles (*QRelationship*). Les relations entre qualités sont indiquées par une valeur d'impact (faible, moyen ou fort) et par une valeur de direction (parallèle ou opposé).

2. *Une politique QoS.* Le client d'un service précise les politiques de qualité de services qu'il souhaite avoir. Les fournisseurs de services proposent plusieurs politiques pour chaque service qu'ils implémentent et qu'ils publient. Les clients et les fournisseurs se basent sur l'ontologie pour décrire, en XML, leurs politiques. Le fournisseur indique par cette politique le niveau d'engagement (garanti, non garanti, *etc*.) de la qualité qu'il publie. Il précise également les valeurs (minimale, maximale) et l'unité de mesure de la qualité spécifiée. Le client indique par cette politique ses préférences en termes de qualités, en précisant les valeurs préférées concernant les qualités en question.

3. *Un algorithme de sélection.* Cette approche propose un algorithme de similarité (*matching algorithm*) permettant de mettre en rapport les politiques QoS des clients avec les politiques QoS des services publiés. L'algorithme est spécifié en utilisant le langage Python (Python,2009).

En résumé, Maximilien et Singh (Maximilien&Singh,2004a) proposent de sélectionner dynamiquement les services de façon à ce qu'ils s'adaptent aux préférences des clients. Ils considèrent la qualité de service comme un attribut qualité mesurable (métrique). Les fournisseurs ainsi que les clients spécifient leurs qualités sous forme de politiques QoS, en se

basant sur une ontologie prédéfinie. Cette approche supporte un algorithme de similarité permettant de sélectionner les services correspondant aux politiques QoS des clients.

Le positionnement de l'approche de Maximilien et Singh, par rapport au cadre de référence proposé plus haut, est présenté au Cadre 3.

Vue Objet
Entité : logiciel
Consensus : Oui
Priorisation : Non
Vue But
But : modéliser, sélectionner
Vue Méthode
Méthode de modélisation : formelle
Méthode d'évaluation : -
Stratégie d'évaluation : -
Méthode de sélection : algorithme de similarité
Contrainte de dépendance : non
Vue Outil
Outil de modélisation : ontologie
Outil de sélection : langage (Python)

Cadre 3. Positionnement de l'approche de Maximilien et Singh

2.4.3 L'APPROCHE DE ZENG ET SES COLLEGUES

Zeng et ses collègues (Zeng et *al.*,2003) proposent une approche orientée qualité pour sélectionner les services élémentaires durant l'exécution du service composite. Ils utilisent les diagrammes d'état (Harel&Naamad,1996) pour spécifier le modèle de processus des services composites. A un diagramme d'état est associé un chemin d'exécution correspondant à une séquence d'états successifs allant de l'état initial jusqu'à l'état final. A un diagramme d'état peut être également associé plusieurs chemins, dans le cas où le diagramme d'état comporte des branchements conditionnels. Un plan d'exécution capture les différentes façons d'exécuter un chemin particulier.

Les auteurs considèrent que la sélection des services doit être faite dynamiquement durant l'exécution du service composite et elle doit prendre en compte les critères de qualité. Particulièrement, la sélection doit considérer les contraintes globales ; *i.e.* la qualité du service composite. Par exemple, la contrainte globale d'un service composite consiste à minimiser sa durée d'exécution. Le processus de sélection proposé est composé de deux étapes principales, à savoir : (1) un modèle de qualité de service Web ; et (2) une sélection de service Web basée sur la qualité.

1. *Un modèle de qualité de services Web.* Zeng et ses collègues (Zeng et *al.*,2003) proposent un modèle de qualité informel qui se base sur les critères de qualité (*i.e.* des

propriétés non-fonctionnelles) correspondant au prix d'exécution, à la durée d'exécution, à la fiabilité, à la disponibilité, et à la réputation. Le but est de guider la composition de services à l'exécution. La qualité de service est exprimée par un vecteur qualité, composé de l'ensemble des critères de qualité cités préalablement. Chaque critère est décrit par son nom, une unité, une description succincte, et optionnellement une règle de calcul de la façon de réaliser la mesure. Par exemple, les critères de qualités prix d'exécution (P), durée d'exécution (D) et fiabilité (F) d'une opération « *op* » d'un service élémentaire « *s* » sont définis comme suit : le prix d'exécution, noté *P(op,s)*, est le coût que l'utilisateur du service doit payer pour l'exécution de cette opération. La durée d'exécution, notée *D(op,s)*, mesure le temps, en secondes, entre le moment où la requête est envoyée et le moment où le résultat est reçu. Elle est calculée en utilisant la formule *D(op,s) = Tprocess(op,s) + Ttransmission(op,s)*, indiquant que la durée d'exécution est la somme du temps de traitement (*Tprocess(op,s)*) et le temps de transmission (*Ttransmission(op,s)*. La fiabilité d'un service, notée *F(s)*, correspond à la probabilité que la réponse à la requête soit fournie dans un temps maximal prédéfini. Elle est calculée en se basant sur les données historiques concernant les anciennes invocations du service à travers la formule *F(s) = N(s)/K*. *N(s)* représente la quantité de temps où le service a été délivré avec succès durant un temps maximal prédéfini et *K* correspond au nombre total d'invocations. Ainsi, le vecteur qualité d'un service élémentaire correspond à :

$$Q(s)=\{Q_{prix}(s), Q_{durée}(s), Q_{fiabilité}(s), Q_{disponibilité}(s), Q_{réputation}(s)\}$$

Les critères de qualités cités préalablement sont utilisés pour évaluer la qualité globale du service composite « *sc* ». L'évaluation globale est faite à travers les fonctions d'agrégation concernant les critères de qualité considérés plus haut. Par exemple, le prix d'exécution (*P(sc)*), la durée d'exécution *D(sc)* et la fiabilité (*F(sc)*) d'un service composite sont définis comme suit : le prix d'exécution d'un service composite est la somme des prix d'exécution de chaque service le composant, à savoir $P(sc) = somme\ P(op_i,s_i)/i=1..N$. La durée d'exécution d'un service composite est calculée, en utilisant l'algorithme du chemin critique (*CPA-Critical Path Algorithm*) (Pinedo,2001), comme suit: $D(sc) = CPA\ (D(op_1, s1),\ldots, D(op_N,s_N))$. La fiabilité d'un service composite est le produit $F(sc) = \Pi(e^{F(si)*z})/i=1..N$. Dans ce cas, le vecteur qualité d'un plan d'exécution (noté P) associé à un service composite est :

$$Q(P)=\{Q_{prix}(P), Q_{durée}(P), Q_{fiabilité}(P), Q_{disponibilité}(P), Q_{réputation}(P)\}$$

2. *Une sélection de service Web basée sur la qualité.* Les services composites sont modélisés par des diagrammes d'état. Plusieurs chemins d'exécution peuvent être associés au service composite. Plusieurs plans d'exécution sont associés à un chemin

d'exécution, en allouant des services Web différents. Les auteurs proposent de résoudre le problème de sélection de services comme suit :

a. à chaque service Web est associé un vecteur qualité ;
b. un chemin d'exécution ou un ensemble de chemins d'exécution est généré, ainsi que les plans d'exécution correspondants. Le vecteur qualité de chaque plan est calculé ;
c. sélectionner le plan d'exécution optimal, en utilisant :
    i. dans le cas d'un seul chemin d'exécution, la sélection du plan d'exécution optimal se base sur les techniques de *Multiple Attribute Decision Making* (MADM) (Köksalan&Zionts,2000), particulièrement la technique *Simple Additive Weighting* (SAW) (Hwang&Yoon,1981) est utilisée.
    ii. dans le cas de plusieurs chemins d'exécution, la sélection du plan d'exécution optimal se base sur les techniques de programmation linéaire (Karloff,1991). Cette approche est implémentée par le prototype SELF-SERV (Sheng et *al.*,2002).

Zeng et ses collègues représentent la priorité en associant des poids à chaque critère de qualité participant au processus de sélection afin de gérer la priorisation.

En résumé, Zeng et ses collègues (Zeng et *al.*,2003) proposent de sélectionner les services élémentaires durant l'exécution du service composite. Ils considèrent la qualité de service comme un vecteur qualité, associé à chaque service élémentaire. Le vecteur qualité est composé de quelques critères non-fonctionnels, tels que le coût et la durée d'exécution. Un ensemble de fonctions d'agrégation est proposé pour calculer la qualité globale du service composite. Les auteurs sélectionnent le service composite optimal en se basant sur deux techniques, à savoir SAW et la programmation linéaire, et en gérant la priorisation en optant pour un système de poids.

Le positionnement de l'approche de Zeng et ses collègues, par rapport au cadre de référence proposé plus haut, est récapitulé au Cadre 4.

Vue Objet
Entité : logiciel
Consensus : Non
Priorisation : Oui
Vue But
But : modéliser, évaluer, sélectionner
Vue Méthode
Méthode de modélisation : informelle
Méthode d'évaluation : quantitative
Stratégie d'évaluation : globale
Méthode de sélection : {MADM,PL}
Contrainte de dépendance : non
Vue Outil
Outil de modélisation : vecteur
Outil de sélection : technique (SAW), outil (SELF-SERV)

Cadre 4. Positionnement de l'approche de Zeng et ses collègues

2.4.4 L'APPROCHE D'AIELLO ET GIORGINI

Aiello et Giorgini (Aiello&Giorgini,2004) utilisent la méthode de développement orientée agent TROPOS (Mylopoulos&Castro,2000) afin de développer des systèmes à base de services, et particulièrement afin de modéliser les exigences de qualité de service.

Aiello et Giorgini considèrent que la qualité de service est l'ensemble de propriétés non-fonctionnelles, telles que la sécurité, la performance et la disponibilité, ainsi que tout paramètre client pouvant être modélisé comme propriété non-fonctionnelle du service. Dans la perspective de raisonner sur la qualité, les techniques d'analyse de but de TROPOS (Giorgini et *al.*,2004) (Sebastiani et *al.*,2004) sont utilisées.

Cette approche est composée de deux étapes à savoir : (1) l'expression des besoins des utilisateurs ; et (2) le raisonnement sur la qualité.

1. *L'expression des besoins des utilisateurs.* Les auteurs proposent d'exprimer les besoins des utilisateurs par des buts. Deux types de but sont identifiés : les buts (*i.e. hard goals)*, relatifs aux exigences fonctionnelles, et les buts qualité (*i.e. soft goals)*, relatifs aux exigences non-fonctionnelles (ou de qualité). Les premiers sont ceux dont la satisfaction est objective ; ils sont dits réalisables. Les derniers sont ceux dont la satisfaction est subjective ; ils sont dits partiellement satisfaisables (*i.e. satisficing*) (Lamsveerde,2001) (Letier&Lamsweerde,2004). Le processus de décomposition de buts est effectué jusqu'à atteindre les sous-buts pouvant être satisfaits par une fonctionnalité du système, appelés tâches ou plans. Aiello et Giorgini considèrent les tâches comme étant les services du futur système. Ces tâches permettent de réaliser des buts et de satisfaire partiellement des buts qualité. Le processus de décomposition de buts est représenté sous forme de graphe :

les nœuds de l'arbre représentent les buts et les tâches, et les arcs sont les relations entre les nœuds. Les relations peuvent être des relations de décomposition de type « *Et/Ou* » ou des relations de type « *contribution* ». Les sous-buts, reliés par une relation « *Et* » avec le but parent, doivent tous être réalisés afin que le but parent soit réalisé également. Dans le cas où les sous-buts sont reliés par une relation « *Ou* », il suffit qu'un seul sous-but soit réalisé pour que le but parent soit réalisé également. Les relations de type contribution permettent de montrer l'impact, pouvant être positif (+), très positif (++), négatif (-) ou très négatif (--), d'une tâche (ou d'un but) sur la satisfaction partielle d'un but qualité. Les relations de contribution sont présentées dans (Giorgini et *al.*,2004) et quelques unes d'entre elles sont résumées dans ce qui suit : Soit G_1 un nœud parent et G_2 un nœud fils. On considère que G_2 contribue *positivement* (ou négativement) à la satisfaction de G_1. Si G_2 est satisfait alors il existe *une évidence* que G_1 soit satisfait (ou rejeté), mais si G_2 est rejeté alors on ne peut rien dire de la possibilité de rejet (ou de satisfaction) de G_1. Dans le cas où G_2 contribue *très positivement* (ou très négativement) à la satisfaction de G_1, si G_2 est satisfait alors il existe *une forte évidence* que G_1 soit satisfait (ou rejeté), mais si G_2 est rejeté alors on ne peut rien dire de la possibilité de rejet (ou de satisfaction) de G_1.

2. *Raisonnement sur la qualité.* Les auteurs proposent de raisonner sur la qualité afin de pouvoir décider des services qui répondent au mieux aux besoins de qualité des utilisateurs. Les auteurs proposent des axiomes permettant d'effectuer un raisonnement qualitatif ou quantitatif, selon le type de contribution pouvant être également qualitatif ou quantitatif.

Le raisonnement qualitatif repose sur des concepts associés à la satisfaction et au rejet d'un but qualité, à savoir *Full Satisfaction* (FS), *Full Denied* (FD), *Partial Satisfaction* (PS), ou encore *Partial Denied* (PD). FS signifie qu'il existe une *évidence forte* que le but soit *satisfait*. FD signifie qu'il existe une *évidence forte* que le but soit *rejeté*. PS signifie qu'il existe une *évidence partielle* que le but soit *satisfait*. PD signifie qu'il existe une *évidence partielle* que le but soit *rejeté*. Les valeurs de satisfaction des buts peuvent être FS, FD, PS, et PD. Plusieurs axiomes sont proposés dans (Aiello&Giorgini,2004). Ils permettent de guider la propagation de la satisfaction des buts à travers l'arbre des buts. Deux exemples d'axiome sont présentés au Tableau 3. Soit G_1 un nœud parent et G_2 un nœud fils. Le premier axiome montre que dans le cas d'une relation de contribution positive, la satisfaction partielle du nœud fils, G_2, implique une satisfaction partielle du nœud parent, G_1. Le deuxième axiome montre que dans le cas d'une relation de contribution très positive, la satisfaction totale du nœud fils, G_2, implique une satisfaction

totale du nœud parent, G_1 et la satisfaction partielle du nœud fils, G_2, implique une satisfaction partielle du nœud parent, G_1. L'algorithme de propagation qualitative et son implémentation sont présentés dans (Giorgini et *al.*,2004). Le principe de l'algorithme est d'assigner des valeurs initiales aux tâches et de propager ces valeurs, du bas vers le haut, en se basant sur les axiomes de raisonnement qualitatif.

Tableau 3. Exemple d'axiome

Symbole	Axiomes
$G_2 \xrightarrow{+} G_1$	PS (G_2) → PS (G_1)
$G_2 \xrightarrow{++} G_1$	FS (G_2) → FS (G_1) PS (G_2) → PS (G_1)

En ce qui concerne le raisonnement quantitatif, Aiello et Giorgini (Aiello&Giorgini,2004) considèrent que le rejet (ou la satisfaction) partiel (partielle) (PS, PD) présentés précédemment est vague. Par exemple, deux services peuvent permettre une satisfaction partielle d'un but qualité, mais l'un deux peut avoir une contribution plus importante que l'autre sur la satisfaction de ce but. Pour cela, il s'agit d'attribuer des valeurs numériques aux valeurs qualitatives des relations de contribution présentées au Tableau 3, permettant ainsi de différencier ces valeurs qui sont qualitativement similaires. Aiello et Giorgini (Aiello&Giorgini,2004) proposent également, en se basant sur d'anciens travaux (Fante,2004)(Giorgini et *al.*,2004), un ensemble d'axiomes permettant de faire un raisonnement quantitatif sur la satisfaction des buts ainsi qu'un algorithme de propagation quantitative.

En résumé, Aiello et Giorgini (Aiello&Giorgini,2004) utilisent TROPOS afin de modéliser la qualité de service comme un but qualité que les services doivent satisfaire partiellement. Ils considèrent la qualité de service comme étant l'ensemble de propriétés non-fonctionnelles, ainsi que tout paramètre client pouvant être modélisé comme propriété non-fonctionnelle du service. Les auteurs de cette approche proposent un ensemble d'axiomes permettant d'évaluer, par propagation, la satisfaction des buts.

Le positionnement de l'approche d'Aiello et Giorgini, par rapport au cadre de référence proposé plus haut, est présenté au Cadre 5.

Vue Objet
Entité : applicatif
Consensus : non
Priorisation : non
Vue But
But : modéliser, évaluer
Vue Méthode
Méthode de modélisation : formelle
Méthode d'évaluation : qualitative
Stratégie d'évaluation : locale
Méthode de sélection/configuration : -
Contrainte de dépendance : non
Vue Outil
Outil de modélisation : langage orienté but
Outil de sélection : -

Cadre 5. Positionnement de l'approche d'Aiello et Giorgini

2.4.5 L'APPROCHE DE PENSERINI ET SES COLLEGUES

A l'instar d'Aiello et Giorgini, Penserini et ses collègues (Penserini et *al.*,2006a) utilisent également la méthode de développement orientée agent TROPOS (Mylopoulos&Castro,2000) afin de couvrir entièrement le cycle de développement des applications à base de services : à partir de l'ingénierie des besoins, regroupant les besoins fonctionnels et les besoins non fonctionnels, à l'identification des services satisfaisant ces besoins.

L'approche est composée de quatre étapes à savoir : (1) expression des besoins des utilisateurs ; (2) identification des capacités ; (3) identification des préférences des utilisateurs ; et (4) évaluation des capacités par rapport aux préférences des utilisateurs.

1. *Expression des besoins des utilisateurs.* Les auteurs proposent d'exprimer les besoins des utilisateurs par des buts. L'analyse des besoins correspond alors à l'identification des buts et l'exploration des diverses alternatives permettant de réaliser ces buts. Le processus de décomposition de buts, tel que présenté dans l'approche d'Aiello et Giorgini (*cf.* Section 2.4.4), met en évidence un ensemble d'*alternatives*. Une alternative représente un sous-arbre de buts permettant de réaliser le but racine.
2. *Identification des capacités.* Penserini et ses collègues (Penserini et *al.*,2006a) proposent de modéliser les services sous la forme de *capacité*. La capacité permet de réaliser des buts et de satisfaire partiellement des buts qualité, lesquels correspondent à la qualité de services. Une capacité est un couple <*compétences*, *opportunités*>. Une compétence est la condition nécessaire pour réaliser un but. Elle est décrite à l'aide de la relation «*means-end*» entre un but et un plan. Une opportunité est la condition suffisante pour réaliser un but ; c'est le contexte permettant de déclencher le plan. Elle est décrite à

l'aide des concepts de plan, de but qualité, de métrique et de contraintes temporelles. Une métrique représente la valeur de la relation de contribution entre un plan et un but qualité, la métrique ∈ {+, -, ++, --}. Ainsi, la capacité est définie comme suit: *Cap = <means_end(plan, but), ∪i contribution (plan, but qualité, métrique), {contraintes_du_domaine}>*. Les capacités sont associées aux alternatives. Par conséquent, une alternative peut être une composition de capacités.

3. *Identification des préférences des utilisateurs.* Les préférences des utilisateurs sont le moyen permettant d'identifier les besoins des utilisateurs en termes de buts fonctionnels à atteindre, ainsi que l'ensemble des besoins non fonctionnels associés. Les auteurs de cette approche (Penserini et *al.*,2006a) se basent sur la formalisation des préférences proposée par Penserini et Mylopoulos (Penserini&Mylopoulos,2005). Ils utilisent le concept de *initial service need* (*isn*), défini par : $isn : < (goal_1, but\ qualité_{1.1}, but\ qualité_{1.2},\ldots),\ldots, (goal_m, but\ qualité_{m.1}, but\ qualité_{m.2},\ldots)>$.

4. *Evaluation des capacités par rapport aux préférences des utilisateurs.* Les capacités sont étudiées par rapport aux préférences des utilisateurs. L'objectif de cette évaluation est de déterminer les couples <*capacités*, *préférences*> qui ont le plus de similitudes. Pour satisfaire ces préférences, des capacités particulières peuvent être favorisées ou défavorisées (grâce aux métriques). A partir d'une préférence d'un utilisateur (*isn*), il s'agit de trouver les capacités (<*compétences*, *opportunités*>) permettant de répondre au mieux à cette préférence. Celle-ci peut être réalisée par une capacité ou par un ensemble de capacités correspondant au service composite.

(Penserini et *al.*,2006b) propose d'utiliser l'architecture dirigée par les modèles (*MDA – Model Driven Architecture*) pour développer les capacités. Tout d'abord les capacités sont décrites sans tenir compte des caractéristiques technologiques, en utilisant le méta-modèle des diagrammes d'interactions et d'activités d'AUML. Ensuite, l'implémentation des capacités est effectuée en se basant sur le méta-modèle JADE (*Java Agent Development Framework*). Des règles de transformation sont ensuite spécifiées afin de permettre le passage d'AUML à JADE.

En résumé, Penserini et ses collègues (Penserini et *al.*,2006a) utilisent TROPOS afin de modéliser la qualité de service comme un but qualité que les services doivent satisfaire partiellement. Ils considèrent que la qualité de service est l'ensemble de propriétés non-fonctionnelles associées aux services, ainsi que toute qualité définie par le client. Penserini et ses collègues proposent d'adapter les services, modélisés sous forme de capacité, aux préférences des utilisateurs. Les capacités sont des plans réalisant des buts et satisfaisant

partiellement des buts qualité. Les préférences correspondent aux buts et aux buts qualité que les utilisateurs souhaitent réaliser.

Le positionnement de l'approche de Penserini et ses collègues, par rapport au cadre de référence proposé plus haut, est présenté au Cadre 6.

Vue Objet
Entité : applicatif
Consensus : Oui
Priorisation : Non
Vue But
But : modéliser, évaluer
Vue Méthode
Méthode de modélisation : formelle
Méthode d'évaluation : qualitative
Stratégie d'évaluation : locale
Méthode de sélection/configuration : -
Contrainte de dépendance : non
Vue Outil
Modèle de modélisation : langage orienté but
Outil de sélection : -

Cadre 6. Positionnement de l'approche de Penserini et ses collègues

2.4.6 L'APPROCHE DE HERSSENS ET SES COLLEGUES

Herssens et ses collègues (Herssens et *al.*,2008) proposent une approche permettant de traiter la sélection des services de façon à prendre en compte les préférences qualités des utilisateurs, lesquelles ne peuvent être réalisées simultanément.

Un important problème est souvent rencontré durant la sélection des services (*QoS-aware service selection*), à savoir la gestion des compromis entre les qualités attendues par les utilisateurs, c'est-à-dire dans le cas où les clients spécifient des niveaux de qualité qui ne peuvent être atteints simultanément par les services.

L'approche de Herssens et ses collègues est composée de deux parties principales à savoir : (1) un modèle de qualité de services ; et (2) une sélection de services orientée qualité.

- *Un modèle QoS.* Le modèle de QoS regroupe les informations importantes pour la sélection de service. Le modèle proposé par cette approche se base sur le cadre de référence de l'OMG UML QoS (OMG, 2008). Ce dernier propose plusieurs concepts associés à la qualité de service, tels que *QoS Characteristics*, *QoS Categories* et *QoS Value*. La qualité de service (*QoS Characteristics*) est définie par ces auteurs comme des critères mesurables associés aux services, tels que le temps de réponse et la latence, et elle possède des valeurs (*QoS Value*). Elle est regroupée en une qualité abstraite (*QoS Categories*), telle que la performance et la sécurité. Ce modèle est utilisé

par les clients pour exprimer leurs attentes en termes de QoS et par les fournisseurs de services pour décrire les QoS de leurs services. Dans la perspective de faire une séparation explicite entre les exigences des utilisateurs et la publication des fournisseurs, Herssens et ses collègues proposent de diviser le modèle OMG en deux : le méta-modèle du fournisseur et celui de l'utilisateur. Ils proposent également d'étendre le modèle OMG en rajoutant quelques concepts, tels que : (i) la priorité afin de préciser explicitement les qualités prioritaires demandées par les utilisateurs ; (ii) les préférences des utilisateurs concernant les valeurs de qualité en les triant par ordre de précédence ; et (iii) la dépendance entre les qualités afin de préciser si les qualités dépendent positivement (ou négativement) les unes des autres. Les priorités et les préférences concernent le méta-modèle de l'utilisateur et les dépendances concernent le méta-modèle du fournisseur. Une description complète de l'extension du modèle de l'OMG est proposée dans (Jureta et *al.*,2008).

- *Une sélection de services orientée qualité*. Le processus de sélection se base sur le modèle de QoS, décrit plus haut, afin d'affiner la découverte de services ayant la même fonctionnalité. Cette approche propose d'utiliser une technique multicritère d'aide à la décision (MCDA – *Multi Criteria Decision Analysis*), particulièrement la technique par la logique floue (*Fuzzy MCDA*). Cette technique permet d'ordonner les services candidats afin de déterminer le service optimal, tout en prenant en compte les divers critères de qualité. Cette technique gère les relations entre les qualités, ainsi que les priorités affectées à celles-ci.

En résumé, Herssens et ses collègues (Herssens et *al.*,2008) proposent de sélectionner les services en prenant en compte les qualités qui ne peuvent être satisfaites simultanément. Ils considèrent la qualité de service comme un critère de qualité mesurable (métrique). Les fournisseurs ainsi que les clients spécifient leurs qualités en se basant sur le modèle de qualité proposé. Les auteurs sélectionnent le service optimal en se basant sur la technique *fuzzy* MCDA, tout en gérant la priorisation des qualités.

Le positionnement de l'approche de Herssens et ses collègues, par rapport au cadre de référence proposé plus haut, est récapitulé au Cadre 7.

Vue Objet
Entité : logiciel
Consensus : Oui
Priorisation : Oui
Vue But
But : modéliser, évaluer, sélectionner
Vue Méthode
Méthode de modélisation : semi-formelle
Méthode d'évaluation : quantitative
Stratégie d'évaluation : locale
Méthode de sélection : MCDA
Contrainte de dépendance : non
Vue Outil
Outil de modélisation: UML
Outil de sélection : technique (fuzzy MCDA)

Cadre 7. Positionnement de l'approche de Herssens et ses collègues

2.4.7 L'APPROCHE DE ROHLEDER DANS LES LIGNES DE PRODUITS

Rohleder (Rohleder,2009) propose une approche orientée-but permettant, durant le développement d'une ligne de produits, de gérer la variabilité fonctionnelle et non-fonctionnelle des exigences. En effet, cette approche traite d'une part la variabilité fonctionnelle des exigences en reprenant les travaux de Bennasri (Bennasri,2005). D'autre part, elle modélise la variabilité non-fonctionnelle des exigences en définissant la qualité de service (QoS-Quality of Service) de chaque variante, afin de dériver les configurations (de produit) qui correspondent au mieux au profil qualité du produit souhaité par les clients.

L'approche de Rohleder (Rohleder,2009) est composé de trois parties principales, à savoir : (1) un modèle de variation de la qualité ; (2) un profil qualité du produit ; et (3) un processus de dérivation du produit.

1. *Un modèle QVaM (Quality Variation Model).* Le modèle QVaM permet de capturer la variabilité fonctionnelle (Bennasri,2005) à travers un formalisme de modélisation orienté-but, appelé la Carte (Rolland,1999). Le formalisme de la Carte (Rolland,1999) permet de représenter les exigences fonctionnelles des utilisateurs sous forme de buts à atteindre et de stratégies permettant de réaliser ces buts. La Carte permet de capturer la variabilité fonctionnelle via des relations de type paquet, multi-segment et multi-chemin. Le modèle QVaM permet également de représenter l'impact des exigences non-fonctionnelles sur les variantes. Il représente les exigences non-fonctionnelles par des buts qualité, en adoptant le *NFR-framework* (Mylopoulos et *al.*,1992). Le modèle QVaM identifie la qualité de service (QoS) de chaque variante comme étant l'ensemble des buts qualité, lesquels impactent ces variantes.

2. *Un profil qualité de produit.* Le client peut exprimer ses besoins, lors de la conception du système, afin de trouver la configuration adéquate du produit. Ce besoin concerne un produit particulier, la QoS que le client souhaite satisfaire, ainsi que les valeurs de satisfaction correspondante.
3. *Un processus de dérivation.* Cette approche propose un processus de dérivation du produit final en appliquant des mesures de similarité (Zoukar,2005). Ce processus permet de mettre en rapport le profil qualité du produit, émis par le client, avec la description de la QoS des variantes, définie dans le modèle QVaM. Un outil, dit *PLM Tool Teamcenter*, est développé pour implémenter le profil et le calcul des mesures de similarité.

En résumé, Rohleder (Rohleder,2009) propose de configurer, lors de la conception d'un produit logiciel, les variantes qui correspondent aux exigences fonctionnelles des clients. Elle considère la QoS comme étant l'impact des buts qualité sur les variantes. Les clients spécifient leurs exigences non-fonctionnelles sous forme d'un profil qualité du produit final. Cette approche supporte une analyse de similarité permettant de sélectionner les configurations de produits correspondant au profil qualité.

Le positionnement de l'approche de Rohleder (Rohleder,2009), par rapport au cadre de référence proposé plus haut, est présenté au Cadre 8.

Vue Objet
Entité : intentionnel
Consensus : non
Priorisation : oui
Vue But
But : modéliser, évaluer, configurer
Vue Méthode
Méthode de modélisation : formelle
Méthode d'évaluation : qualitative
Stratégie d'évaluation : locale
Contrainte de dépendance : non
Méthode de sélection : analyse de similarité
Vue Outil
outil de modélisation : modèle orienté but (QVaM)
Outil de sélection : PLM Tool Teamcenter

Cadre 8. Positionnement de l'approche de Rohleder (Rohleder,2009)

2.5 *RESUME DE L'EVALUATION*

Nous avons présenté, dans ce chapitre, les principales définitions concernant la qualité de service, aussi bien dans la communauté SOC que celle de l'ingénierie des exigences. Nous utilisons le cadre de référence (*cf.* Section 2.3) pour étudier les différentes approches traitant

de la qualité de service, dans le domaine de l'ingénierie des services ou de l'ingénierie des exigences. Ce cadre est constitué de quatre vues complémentaires, lesquelles permettent d'étudier la qualité de service en répondant aux questions : quoi, pourquoi, comment et par quel moyen ?. Le cadre de référence (*cf.* Cadre 1) est utilisé pour étudier et comparer sept approches traitant de la qualité de service dans l'ingénierie des services ou un domaine connexe. Le Tableau 4 synthétise les résultats de cette étude.

D'après cette étude, les principales approches d'ingénierie de services, traitant de la qualité de service, sont partagées entre celles qui gèrent la qualité au niveau des services logiciels (Zeng et *al.*,2003) (Maximilien&Singh,2004a) (Herssens et *al.*,2008), d'autres s'intéressent à la qualité au niveau applicatif (Aiello&Giorgini,2004) (Penserini et *al.*,2006a) (Zachos et *al.*,2009), alors que (Rohleder,2009) se focalise sur la qualité de service au niveau intentionnel.

Nous pensons que les services et leur qualité doivent être orientés vers l'utilisateur final de façon à répondre à leurs exigences fonctionnelles et non-fonctionnelles, sans autant omettre la perspective logicielle de ces services et de leurs qualités.

La majorité des approches (Penserini et *al.*,2006a) (Maximilien&Singh,2004b)(Dobson et *al.*,2005) (Rohleder,2009) s'intéressent au problème du consensus qualité, en définissant des modèles permettant d'une part aux fournisseurs de décrire la qualité de leurs service, et d'autre part aux clients de définir leurs besoins.

Nous considérons également qu'il est important d'assurer un consensus qualité entre les différentes parties prenantes, à savoir les fournisseurs et les clients de service. Néanmoins, les ontologies proposées par la communauté SOC restent difficilement compréhensibles à l'utilisateur final, et celles de la communauté RE sont déconnectées des mesures logicielles.

Toutes les approches étudiées dans cette thèse permettent de modéliser la qualité de service. La majorité des approches (Aiello&Giorgini,2004) (Penserini et *al.*,2006a) (Rohleder,2009) (Maximilien&Singh,2004a) (Zachos et *al.*,2009) propose de modéliser la qualité formellement : par des buts qualité pour les trois premières et par des ontologies pour les deux dernières. Zeng et ses collègues (Zeng et *al.*,2003) fournissent une définition informelle de la qualité, tandis que Herssens et ses collègues (Herssens et *al.*,2008) spécifient la qualité de façon semi-formelle. Trois approches seulement (Zeng et *al.*,2003)(Herssens et *al.*,2008) (Rohleder,2009) s'intéressent à la prise en compte des qualités conflictuelles.

Nous pensons que la modélisation de la qualité de service doit être formelle et orientée vers l'utilisateur final et ses exigences non-fonctionnelles. Néanmoins, il est important de considérer la caractéristique d'interdépendance des exigences non-fonctionnelles.

Cette étude a montré que l'évaluation de la qualité de service se fait quantitativement ou qualitativement. Zeng et ses collègues (Zeng et *al.*,2003) suggèrent une évaluation quantitative globale de la qualité des services logiciels composites. Les approches de l'ingénierie des exigences (Penserini et *al.*,2006a) (Aiello&Giorgini,2004) (Rohleder,2009), quant à elles, font une évaluation qualitative locale de la satisfaction partielle des buts qualité.

Nous croyons qu'il est important d'évaluer la qualité de service qualitativement, selon les termes des utilisateurs finaux (« satisfait », « pas satisfait », *etc*.).

Tableau 4. Résumé de l'évaluation des approches

	(Zeng,2003)	(Maximilien,2004)	(Aiello,2004)	(Penserini,2006)	(Herssens,2008)	(SeCSE,2009)	(Rohleder,2009)
OBJET							
Entité	logiciel	logiciel	applicatif	applicatif	logiciel	applicatif	intentionnel
Alignement	non	oui	non	oui	oui	oui	oui
Priorisation	oui	non	non	non	oui	non	oui
BUT							
But	modéliser, évaluer, sélectionner	modéliser, sélectionner	modéliser, évaluer	modéliser, évaluer, sélectionner	modéliser, sélectionner	modéliser, sélectionner	modéliser, évaluer, configurer
METHODE							
Méthode de modélisation	informelle	formelle	formelle	formelle	semi-formelle	formelle	formelle
Méthode d'évaluation	quantitative	-	qualitative	qualitative	-	quantitative	qualitative
Stratégie d'évaluation	globale	-	locale	locale	-	-	locale
Méthode de sélection	MCDA, PL	algorithme de similarité	-	-	MCDA	algorithme de similarité	algorithme de similarité
Contrainte de dépendance	non	non	non	non	non	non	non
OUTIL							
outil de modélisation	vecteur	ontologie	langage orienté but	langage orienté but	UML	ontologie	langage orienté but
outil de sélection	SAW, SELF-SERV	Python	-	JADE	fuzzy MCDA	UCaRE,EDDiE	PLM Tool Teamcenter

La plupart des approches étudiées dans ce rapport (Zeng et *al.*,2003) (Maximilien&Singh,2004a) (Herssens et *al.*,2008) (Zachos et *al.*,2009) (Penserini et *al.*,2006a) utilisent la qualité de service pour faire de la sélection de service. Ces approches sont partagées entre celles (Zeng et *al.*,2003) (Maximilien&Singh,2004a) (Herssens et *al.*,2008) qui font la sélection à l'exécution des services logiciels, et d'autres (Zachos et *al.*,2009) (Penserini et *al.*,2006a) qui font la sélection de services lors de la conception du

système. Ces approches utilisent différentes techniques pour faire de la sélection de services. Il existe celles (Zeng et *al.*,2003) (Herssens et *al.*,2008) qui se basent sur les méthodes multicritères d'aide à la décision, et d'autres (Maximilien&Singh,2004a) (Zachos et *al.*,2009) (Rohleder,2009) qui utilisent des algorithmes de similarité.

Nous pensons qu'il est important de sélectionner et d'exécuter les services orientés vers l'utilisateur final. La sélection doit prendre en compte la caractéristique d'interdépendance des exigences non-fonctionnelles.

La qualité de service est également utilisée pour configurer un système à forte variabilité. En effet, l'objectif est d'exploiter la qualité pour éliminer (ou sélectionner) les variantes qui permettent de respecter les qualités attendues par l'utilisateur. Dans ce contexte, Rohleder (Rohleder,2009) propose de faire de la configuration de ligne de produits. Nous pensons qu'il est intéressant, dans le cadre de l'ingénierie des services, d'exploiter la qualité de service pour configurer des services agrégats contenant plusieurs manières possibles de composer les services, en fonction du contexte qualitatif souhaité par l'utilisateur final. Néanmoins, les travaux dans le domaine des lignes de produit (Bühne et *al.*, 2005)(Djebbi,2011) ont permis de démontrer que des contraintes de dépendance entre variantes existent et qu'elles doivent être prises en compte au niveau du processus de configuration, sinon la configuration de produit résultante peut être invalide ou incohérente au niveau de l'exécution du logiciel.

CHAPITRE 3 : APERÇU DE L'APPROCHE PROPOSEE

L'analyse de l'état de l'art (*cf.* Chapitre 2) a fait ressortir les problèmes énoncés dans cette thèse, à savoir : (i) la discordance conceptuelle entre les exigences (fonctionnelles et non-fonctionnelles) des utilisateurs finaux et les services logiciels et leurs qualités ; (ii) le consensus qualité entre les fournisseurs et les utilisateurs finaux ; et (iii) la sélection et l'exécution des services de haut niveau, en fonction des exigences non-fonctionnelles des utilisateurs finaux ; et (iv) la configuration exécutable des services de haut niveau, dirigée par les exigences non-fonctionnelles des utilisateurs finaux.

Pour résoudre ces problèmes, nous présentons dans ce chapitre un aperçu de l'approche proposée dans cette thèse. Il s'agit d'une méthode de sélection et de configuration intentionnelle des services, en fonction des exigences non-fonctionnelles des agents métier (l'agent métier représentant l'utilisateur final).

3.1 *VUE GLOBALE DE LA METHODE*

La vue globale de l'approche de sélection et de configuration intentionnelle des services que nous proposons est présentée à la Figure 11. Celle-ci est guidée par les exigences non-fonctionnelles des clients et présente quatre éléments importants, à savoir : le référentiel qualité ; le modèle *MiS-q* ; le processus de sélection intentionnelle des services, qui comprend le contexte qualité ; et enfin la configuration intentionnelle de service.

Comme nous avons pu observer dans l'état de l'art, l'établissement d'un *consensus* autour de la définition de la qualité entre les clients et les fournisseurs, ainsi qu'entre les fournisseurs eux-mêmes, est un problème pas encore résolu. Un référentiel qualité entre les différents acteurs est nécessaire afin d'homogénéiser le vocabulaire et la compréhension des qualités par tous les acteurs engagés. Un tel référentiel doit être à la fois compréhensible par les agents métier et permettre la mise en place d'un pont entre cette compréhension de haut niveau et les qualités offertes par les services logiciels. Or, les travaux que nous avons étudiés dans l'état de l'art ne présentent pas ces caractéristiques. Ils se concentrent soit sur la définition d'ontologies souvent trop technique pour les agents métier (Dobson et *al.*,2005), soit sur la

définition ou l'usage de cadres de référence, qui sont beaucoup trop éloignés des services logiciels et de ce qu'ils offrent réellement comme qualité de service.

Nous proposons dans cette thèse, la définition formelle d'un méta-modèle de la qualité, nommé le *référentiel qualité* (correspond à *(1)* de la Figure 11). Ce référentiel se situe dans une position de médiateur entre les différents acteurs engagés, agents et fournisseurs métier, se rapportant directement au problème de consensus entre ces acteurs.

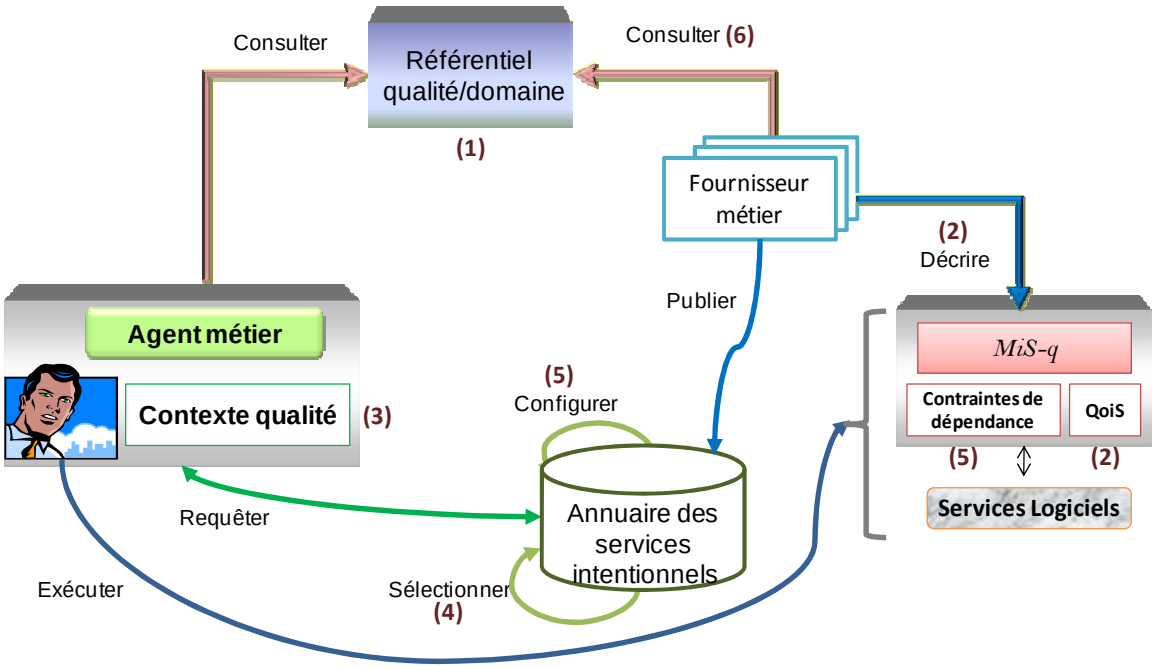

Figure 11. Vue globale de l'approche proposée

Un second problème se dégage de l'état de l'art, à savoir la *discordance conceptuelle* qui existe entre l'expression des exigences des agents métier et la définition des services. D'un côté, les services logiciels sont spécifiés à l'aide de langages, tels que WSDL ou OWL-S, qui sont beaucoup trop techniques pour être compréhensibles par les agents métier. De même pour la définition des QoS que ces services offrent. Celles-ci sont bien souvent décrites de manière quantitative à l'aide des métriques et des mesures, lesquelles sont souvent incompréhensibles par les agents métier (par exemple, un temps de réponse entre *1s* et *1,5s* garantit-il une bonne performance ?). D'un autre côté, certains auteurs (Penserini et *al.*,2006a) ont proposé une définition qualitative des QoS (à travers les buts qualité), subjective mais compréhensible par les agents métier. Cependant, ces définitions restent conceptuelles, utilisables lors de la spécification d'un service, mais déconnectées du service logiciel et de son exécution.

L'architecture iSOA proposée, par Kaabi (Kaabi et *al.*,2004) (Kaabi,2007) et Rolland (Rolland et *al.*,2007) (Rolland&Kaabi,2007), s'est attaquée au problème de discordance conceptuelle, en définissant la notion de service intentionnel, compréhensible par les agents

métier. Un service intentionnel, même s'il est défini autour des buts exprimés par les agents métier, reste opérationnalisable par un service logiciel, ce qui lui permet de réduire le problème de discordance conceptuelle, concernant les exigences fonctionnelles. Cependant, ce problème reste présent dans l'architecture iSOA en ce qui concerne la qualité de service.

Dans cette thèse, nous proposons d'attaquer le problème de discordance conceptuelle concernant les aspects non-fonctionnels, par l'extension de l'architecture iSOA à ces aspects. Nous proposons une définition formelle d'un modèle intentionnel de la qualité de service, nommé *MiS-q* (correspond à *(2)* de la Figure 11). Ce modèle permet, à l'aide du référentiel qualité (élément *(1)* de la Figure 11), de décrire les qualités (élément *QoiS* de la Figure 11) offertes par un service d'un point de vue qualitatif, avec une approche intentionnelle, plus facile à comprendre par les agents métier. Cette description qualitative est associée à une description quantitative, reliées aux métriques propres aux services logiciels.

Cette thèse propose également un processus de sélection intentionnelle des services (élément *(4)* de la Figure 11). Celui-ci comprend la description d'un modèle de contexte qualité (correspond à *(3)* de la Figure 11), qui permet aux utilisateurs d'établir, à l'aide du référentiel qualité, leurs préférences en termes de qualité de service. Le processus de sélection intentionnelle des services permet à un agent métier de sélectionner, dans un annuaire de services intentionnels, ceux qui correspondent au mieux à leurs exigences non-fonctionnelles.

Lorsque le service sélectionné supporte la variabilité dans la manière d'atteindre les buts des agents métier, un processus de configuration intentionnelle des services (correspondant à *(5)* à la Figure 11) est appliqué. Une telle variabilité permet de répondre aux exigences fonctionnelles et non-fonctionnelles des agents de différentes manières, imposant le choix des meilleures variantes au sein même d'un service. De plus, ce processus doit aboutir à une configuration exécutable. L'exécutabilité d'une configuration est souvent associée dans la littérature au respect des contraintes de dépendance (Verma et al.,2005)(Cohen&Krut,2010)(Lutz,2000). Or, le modèle *MiS* (Kaabi,2004)(Kaabi,2007) ne permet pas l'expression de ces contraintes. Nous proposons donc dans cette thèse d'étendre le descripteur des services intentionnels à la prise en compte des contraintes de dépendance entre services. Celles-ci seront exploitées par l'algorithme de configuration (correspondant à *(5)* à la Figure 11).

En outre, nous proposons également, dans cette thèse, un guidage méthodologique (élément *(7)* de la Figure 11) permettant l'identification et la description des qualités et des contraintes de dépendances, liées au service intentionnel. Ce guidage est sensé aider les fournisseurs de service dans la description de leurs services intentionnels.

Toute cette approche concerne le problème de sélection et de configuration des services dédiés à l'utilisateur. Nous croyons que la sélection et la configuration de ces services, qui interagissent directement avec les utilisateurs finaux, doit être guidée par les exigences fonctionnelles et non-fonctionnelles de ces utilisateurs, et non plus par des aspects purement techniques (typiquement les paramètres d'entrée et de sortie).

Nous introduisons chacun de ces éléments dans la section suivante.

3.2 *ELEMENTS DE LA METHODE*

3.2.1 LE META-MODELE DU REFERENTIEL QUALITE

Le problème de consensus qualité est reconnu dans les deux communautés SOC (Maximilien&Singh,2004b) (Dobson et *al.*,2005) (Wang et *al.*,2006) et RE (Penserini et *al.*,2006) (Ma et *al.*,2009), sur lesquelles se positionne ce travail.

Le référentiel qualité (élément *(1)* de la Figure 11), que nous proposons dans cette thèse, représente le méta-modèle de notre méthode. Il fournit, par domaine d'activité, un catalogue de qualité, fournissant une uniformisation des qualités proposées. Il permet, au niveau intentionnel, aux fournisseurs et aux agents métier d'utiliser les mêmes concepts et les mêmes seuils de satisfaction. Le référentiel qualité permet ainsi de garantir le consensus qualité entre les fournisseurs et les agents métier : les fournisseurs peuvent décrire la qualité de leurs services, en alimentant le modèle *MiS-q*. De leur côté, les agents métier peuvent formaliser leurs exigences non-fonctionnelles sous forme de contexte qualité.

Nous proposons que le référentiel qualité soit basé sur les concepts du cadre NFR (*i.e. NFR Framework*) (Chung et *al.*,2000), lequel est une référence dans le traitement des exigences non-fonctionnelles, telles que la sécurité, la performance et la précision. Les exigences non-fonctionnelles sont représentées par des buts qualité à satisfaire partiellement. Le raisonnement sur les buts qualité repose alors sur le concept de *satisficing*. Celui-ci correspond à la définition d'un ensemble de *seuils de satisfaction*, lesquels précisent que les buts qualité sont satisfaits dans des limites acceptables, plutôt que de façon absolue. Un but qualité peut alors être « satisfait », « très satisfait », « peu satisfait » ou « pas du tout satisfait ».

Le référentiel qualité que nous proposons dans cette thèse soutient une décomposition, par domaine particulier, des buts qualités. Dans le cas de buts qualité mesurables ou quantifiables (*e.g.* performance), nous proposons de définir les sous-buts qualité (*e.g.* temps de réponse) obtenus comme des métriques. A travers ces métriques, le référentiel qualité permet

d'associer aux seuils de satisfaction prédéfinis une évaluation quantitative, moins subjective. En effet, les valeurs de ces métriques reflètent la qualité calculée sur les services logiciels. Nous proposons de regrouper ces valeurs en intervalles ou en ensembles de valeurs, nommés classes. Chaque classe correspond alors à un seuil de satisfaction des buts qualité. Le regroupement de tous ces éléments (*i.e.* les buts qualité, les métriques, *etc*.) constitue le référentiel qualité. Il pourrait exister autant de référentiels que de domaines.

Nous pouvons ainsi dire que le référentiel qualité résout le problème de consensus qualité, car :

1. il propose une expression unifiée des exigences non-fonctionnelles et de la qualité des services logiciels entre les différents acteurs de l'architecture (agent et fournisseur métier) ;
2. il permet d'établir la mise en correspondance au niveau intentionnel entre les exigences non-fonctionnelles des agents métier et la qualité du service intentionnel, établie par le fournisseur métier ;
3. il permet d'améliorer la crédibilité des fournisseurs, car tous les fournisseurs d'un domaine se basent sur le même référentiel pour décrire la qualité de leurs services ;
4. il permet de diminuer le risque d'incompatibilité lors de la sélection des services ; et
5. il est un bon moyen de réutilisation des qualités, en évitant aux fournisseurs de construire leurs propres référentiels.

3.2.2 LE MODELE MIS-Q

Le modèle *MiS-q* (élément *(2)* de la Figure 11) correspond à la représentation formelle du service intentionnel, en y intégrant ses qualités et ses contraintes de dépendance.

Le service intentionnel originellement proposé par (Kaabi,2007) est exprimé de façon intentionnelle en mettant en avant les buts que les utilisateurs souhaitent réaliser en exécutant ce service. Le service intentionnel est dédié aux utilisateurs finaux, il permet de diminuer la discordance conceptuelle entre l'énoncé des exigences fonctionnelles de ces utilisateurs et la définition des services logiciels. Néanmoins, le problème de discordance conceptuelle reste persistant, en ce qui concerne la mise en correspondance entre les exigences non-fonctionnelles des utilisateurs et la qualité des services logiciels. De plus, le problème d'exécutabilité de la configuration persiste : le service intentionnel introduit de la variabilité dans la réalisation du but du service, en encapsulant différentes variantes du service. Ainsi, lors de la configuration du service, le choix de variantes d'un service peut alors exiger ou exclure le choix d'autres variantes du même service. Le modèle *MiS-q* (*cf.*

Figure 12) que nous proposons dans cette thèse s'attaque à ces deux problèmes, notamment par la définition aussi bien de la qualité du service intentionnel, que des contraintes de dépendance entre services.

Tel que Bühne et ses collègues (Bühne et *al.*, 2005), nous considérons que la variabilité encapsulée dans le service intentionnel encourage à considérer les dépendances fonctionnelles pouvant exister entre les variantes d'un service agrégat. Ainsi, nous définissons des *contraintes de dépendance* (*cf.* Figure 12) entre services intentionnels. Les dépendances sont valides dans le contexte d'un service agrégat donné, et s'appliquent à des services atomiques ou agrégats. Le modèle *MiS-q* permet de représenter les dépendances *Exclut* et *Exige* : la dépendance Exclut définit que l'exécution d'un service écarte l'exécution d'un autre service, tandis que la dépendance Exige définit que l'exécution d'un service requiert l'exécution d'un autre service.

Comme Rohleder (Rohleder,2009), nous considérons que la qualité de service peut être vue comme l'ensemble des buts qualité que le service permet partiellement de satisfaire. Ainsi, le modèle *MiS-q* permet de représenter, au niveau intentionnel, la *qualité du service intentionnel* (*QoiS-Quality of intentional Service*) que peut offrir les services, et la possibilité de mettre la QoiS à la disposition des agents métier.

Un service intentionnel permet de contribuer, à un certain seuil de satisfaction, à des buts qualité. Le but qualité peut être « satisfait », « très satisfait », « peu satisfait » ou « pas du tout satisfait » une fois le service intentionnel exécuté. Tel que le montre la Figure 12, la QoiS peut être une QoiS simple ou une QoiS globale. La QoiS est dite simple, lorsqu'elle correspond à la qualité d'un seul service intentionnel, contrairement à la QoiS globale qui correspond à la qualité de plusieurs services intentionnels. La QoiS simple est associée au service atomique, tandis que la QoiS globale correspond au service agrégat. La QoiS simple est dérivée à partir des qualités du service logiciel et de ses qualités métier. Ainsi, le modèle *MiS-q*, comme le modèle *MiS*, considère que l'opérationnalisation du service intentionnel atomique est réalisée par un service logiciel. Dans la pratique, un même service intentionnel peut être opérationnalisé par plusieurs services logiciels concurrents. La prise en compte de l'opérationnalisation du service atomique sera faite au niveau de la méthodologie (élément (6) de la Figure 11). Concernant la QoiS globale, nous proposons une évaluation de qualité qui dépend de la nature du service. En effet, le service agrégat est représenté par une décomposition de but *Et/Ou*. L'évaluation de la qualité d'un tel service dépend alors de cette relation existant entre ses services. De cette manière, la QoiS d'un service agrégat (*i.e.* la

QoiS globale) est elle-même évaluée, récursivement, en fonction des QoiS composantes, lesquelles peuvent être des QoiS globale ou simple.

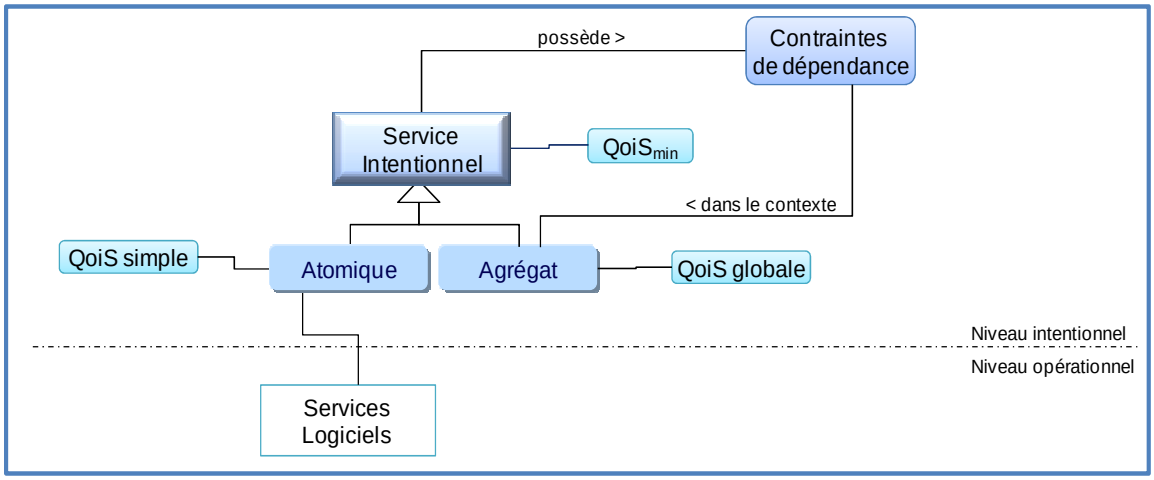

Figure 12. Vue globale du modèle *MiS-q*

La QoiS globale, telle que définie dans le modèle *MiS-q*, n'est pas directement exploitable par le processus de sélection de services, à cause des différentes variations qui peuvent exister à l'intérieur d'un même service intentionnel. Grâce au service à variation, le service intentionnel introduit de la variabilité dans la manière d'atteindre le but du service et de satisfaire des buts qualité. De manière similaire à une ligne de produits logiciels, la variabilité du service intentionnel permet de mettre en évidence les caractéristiques communes à tout processus réalisant un but, tout en rendant explicite les variations possibles. Ces variations concernent non seulement les différentes manières de réaliser le but du service, mais également les différentes manières de satisfaire partiellement les buts qualité. Ces dernières permettent, dans le cadre d'une composition de services, de répondre aux exigences fonctionnelles et non-fonctionnelles des clients de différentes manières. Il faut alors prendre en considération cette composition et ses qualités lors de la description et de la publication d'un service intentionnel.

Pour cela, nous proposons d'évaluer la qualité dite $QoiS_{min}$ (*cf.* Figure 12). Celle-ci se base sur le principe de propagation du seuil minimal de satisfaction des buts qualité. Comme son nom l'indique, la $QoiS_{min}$ fournit la qualité minimale garantie par le service agrégat, sachant que ce dernier pourrait fournir une meilleure qualité en fonction de ses configurations. La $QoiS_{min}$ est utilisée par le processus de sélection intentionnelle pour différencier les différents services agrégats. Nous proposons de calculer la qualité moyenne ($QoiS_{moy}$), afin de comparer les configurations exécutables d'un même service agrégat. Un

ensemble de règles est proposé pour évaluer la QoiS simple, la QoiS globale, et la $QoiS_{min}$ et la $QoiS_{moy}$.

3.2.3 LA METHODOLOGIE

La méthodologie a pour objectif de fournir aux fournisseurs métier des guides méthodologiques, afin d'identifier et de décrire, selon les termes du modèle *MiS-q*, les différentes variantes métier constituant un service de haut niveau, à savoir le service intentionnel.

Dans le modèle *MiS-q*, la dimension qualitative du service est également orientée but. Elle est fournie par la qualité du service intentionnel (*QoiS-Quality of intentional service*). Notre choix de dériver la qualité de service à partir d'un modèle métier orienté but a deux motivations principales, à savoir : (i) les modèles orientés buts utilisent un langage compréhensible par les agents métier, contrairement aux modèles de qualité de service qui exigent des connaissances techniques ; et (ii) les modèles orientés buts modélisent le « pourquoi » du système, ce qui facilite la sélection des services, car à partir des besoins de l'agent métier (ses intentions), il peut identifier les services qui satisfont ses besoins.

Selon Kaabi (Kaabi,2007), le modèle de la Carte (Rolland&Prakash,2000) est adapté à la découverte des services. Il supporte la découverte de la variabilité des services. Chaque combinaison possible de buts et de stratégies identifie une variante de services possible pour atteindre le but du service global.

Nous utilisons également le modèle de la Carte pour exprimer les exigences fonctionnelles et non-fonctionnelles des agents métier. Les exigences fonctionnelles sont représentées par la carte des besoins, laquelle est un graphe orienté dans lequel les nœuds sont des buts à atteindre et les arcs des stratégies définissant la manière de réaliser ces buts. Les exigences non-fonctionnelles sont représentées par des liens de contribution entre la carte des besoins et les buts qualité. Dans le cadre de l'architecture iSOA, nous proposons aux fournisseurs métier (élément (*6*) de la Figure 11) une méthodologie leur permettant, à partir du référentiel qualité (élément (*1*) de la Figure 11), de définir les buts qualité que la carte des besoins va satisfaire partiellement. Chaque section opérationnalisable de la carte des besoins peut alors contribuer à la satisfaction partielle des buts qualité. Les liens de contributions sont utilisés pour mettre en évidence la contribution d'une section pour (« + » ou « ++ ») ou contre (« - » ou « -- ») la satisfaction d'un but qualité particulier.

Les services et leurs qualités sont en relation avec les buts des agents métier. Nous proposons alors aux fournisseurs métier une méthodologie leur permettant, à partir de la carte

des besoins et des liens de contribution reliant la carte aux buts qualité, d'identifier les services et leur QoiS. A l'issu de cette étape la description du service agrégat de haut niveau est opérationnelle jusqu'au niveau logiciel.

Enfin, pour garantir l'exécutabilité des configurations dérivées à partir de cet agrégat, des contraintes de dépendance entre variantes de services peuvent être identifiées et décrites au niveau *MiS-q*. Le fournisseur métier peut ainsi exposer un descripteur intentionnel de service agrégat de haut niveau dont la configuration pourra être dirigée par les qualités et exécutable.

3.2.4 LA SELECTION INTENTIONNELLE DES SERVICES

La sélection des services consiste à affiner la découverte des services, en prenant en compte les critères de qualité. Nous proposons un processus de sélection intentionnelle des services, lequel est centré sur le client. Le processus de sélection intentionnelle des services correspond à un mécanisme d'exécution intentionnelle de ces services, lequel cherche à guider l'agent métier dans le choix des services répondant au mieux à son contexte qualité.

Le processus de sélection intentionnelle que nous proposons dans cette thèse est organisée en deux parties : (1) le modèle du contexte qualité (élément (*3*) de la Figure 11) qui correspond au profil de l'agent métier. Ce contexte permet de formaliser les exigences de l'agent métier, particulièrement ses préférences en termes de qualité ; et (2) la sélection intentionnelle des services (élément (4) de la Figure 11), laquelle consiste à choisir parmi les services des différents fournisseurs, ceux qui correspondent au mieux au contexte qualité de l'agent métier.

3.2.4.1 *LE MODELE DU CONTEXTE QUALITE*

Nous proposons de formaliser les exigences de l'agent métier dans un *contexte qualité* (élément *(3)* de la Figure 11). Le contexte qualité (*cf.* Figure 13) est composé d'un but fonctionnel (un *hard goal*), dit but (*cf.* Figure 13), que l'agent métier désire réaliser et d'un ensemble de buts qualité qu'il souhaite partiellement satisfaire. Pour chaque but qualité, l'agent métier précise le seuil de satisfaction au-delà duquel il accepte toute satisfaction, ainsi que la priorité qu'il associe à chaque qualité. Cette dernière est précisée afin d'exprimer une préférence lorsque les qualités sont conflictuelles. Par exemple, la sécurité et la performance sont des qualités conflictuelles, car un service fournissant une forte sécurité (en implémentant, par exemple, plusieurs algorithmes de cryptage) diminue forcément la performance de ce service.

Dans le cadre de l'architecture iSOA, nous proposons que l'agent métier se base également sur le référentiel qualité, d'une part, pour décider des qualités qu'il souhaite satisfaire, et d'autre part, pour vérifier la sémantique des seuils de satisfaction, afin qu'il définisse les qualités et les seuils qui lui conviennent réellement. Ceci permet de diminuer les risques d'incompatibilité lors du processus de sélection intentionnelle des services.

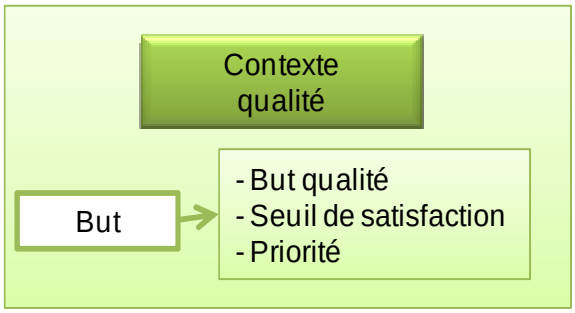

Figure 13. Vue globale du modèle du contexte qualité

3.2.4.2 *LA SELECTION INTENTIONNELLE DES SERVICES*

La sélection intentionnelle des services (élément *(4)* de la Figure 11) consiste à choisir parmi les services des différents fournisseurs ceux qui correspondent au mieux au contexte qualité de l'agent métier. En effet, plusieurs fournisseurs métier peuvent fournir des services de même ligne, c'est-à-dire réalisant le même but. Néanmoins, ces services offrent des QoiS différentes : ils peuvent satisfaire partiellement plusieurs buts qualité avec des seuils de satisfaction différents, ou même ne satisfaire aucun but qualité.

On est confronté à un problème de décision pour lequel il existe un nombre fini de services intentionnels, lesquels réalisent le but de l'agent métier. Il existe également plusieurs buts qualité, lesquels correspondent aux exigences non-fonctionnelles que l'agent métier souhaite satisfaire. Généralement, aucun de ces services n'arrive à satisfaire complètement tous les buts qualité, surtout lorsque ces derniers sont conflictuels. Il s'agit alors de chercher un compromis parmi les services intentionnels existants, en combinant les différents buts qualité, afin de pourvoir choisir ceux qui se rapprochent le plus du contexte qualité de l'agent métier.

Comme (Zeng et *al.*,2003) (Wang et *al.*,2006) (Herssens et *al.*,2008) (Baryannis et *al.*, *2008*) (Ma et *al.*,2009), nous pensons que les méthodes multicritères d'aide à la décision sont un bon moyen de résoudre ce type de problème. Nous proposons d'utiliser la méthode multicritères d'aide à la décision TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*), laquelle est développée par Hwang et Yoon (Hwang&Yoon,1981) et a fait l'objet de nombreuses applications (Schinas,2007) (Yamamoto&Saeki,2008). La méthode TOPSIS permet de sélectionner, dans un espace géométrique, les alternatives ayant la plus

courte distance d'une solution hypothétique, considérée comme la meilleure solution, et la plus grande distance d'une seconde solution hypothétique, représentant la pire solution. La distance Euclidienne est utilisée pour évaluer la proximité (ou la similitude) relative par rapport à la meilleure et à la pire solution.

Selon Shinas (Schinas,2007), la méthode TOPSIS offre plusieurs avantages aux décideurs, particulièrement au non académiciens, lesquels ne considèrent pas les problèmes théoriques : la méthode TOPSIS fournit une solution euclidienne, c'est-à-dire facile à concevoir, et tous les calculs se font dans un temps polynomial. Les décideurs peuvent comprendre plus facilement la déviation d'une solution par rapport à une meilleure solution ou à une pire solution, avec une méthode qui est plus simple à mettre en œuvre que d'autres comme Electre (Benayoun, et *al.*,1966). Par ailleurs, nous avons aussi opté pour TOPSIS pour sa flexibilité, car il est possible de la personnaliser. En effet, au lieu de sélectionner les alternatives ayant la plus courte distance de la meilleure solution, nous proposons d'adapter TOPSIS, afin de sélectionner les services ayant la plus courte distance du contexte qualité de l'agent métier. Bien entendu, ce choix n'est pas le seul applicable et il peut être discuté sans remettre en cause l'intérêt de la méthode TOPSIS.

Par ailleurs, les services agrégats sélectionnés peuvent encapsuler différentes variantes qui peuvent fournir une qualité plus intéressante que celle proposée par le service agrégat (la sélection se fait sur la qualité minimale). Nous parlerons alors de configuration de service.

3.2.4.3 *LA CONFIGURATION INTENTIONNELLE DES SERVICES*

Le service agrégat, et particulièrement le service à variation, est composé de plusieurs variantes, lesquelles réalisent le même but (celui du service agrégat), mais fournissent potentiellement des QoiS différentes. Une fois sélectionné, le service agrégat offre alors différentes alternatives, des variations proposant différentes qualités de service, qui peuvent ne pas répondre totalement aux exigences non-fonctionnelles de l'agent métier. Par analogie à la configuration de produits (Bühne et *al.*,2005), nous proposons un processus de configuration intentionnelle du service agrégat (élément *(5)* de la Figure 11). Celui-ci consiste dans un premier temps, à sélectionner les variantes des services au mieux de la qualité demandée. Dans un deuxième temps, il s'agit de considérer les contraintes de dépendance entre services, afin de valider l'exécutabilité des différentes configurations obtenues. Ce processus permet d'unifier à travers *MiS-q*, la sélection dirigée par les qualités des variantes métier et des variantes logicielles opérationnalisant les variantes métier choisies.

Nous proposons d'utiliser, d'une part, la méthode TOPSIS pour configurer le service agrégat au plus proche du contexte qualité de l'agent métier. D'autre part, nous exploitons les contraintes de dépendances pour valider l'exécutabilité des configurations obtenues.

Par ailleurs, nous avons développé le prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*), afin d'implémenter la méthode TOPSIS, adaptée à la sélection et à la configuration intentionnelle des services.

3.3 *EN RESUME*

L'approche que nous proposons dans cette thèse, se positionne alors de la façon suivante par rapport au cadre de référence (*cf.* Cadre 9).

Vue Objet
Entité : intentionnel
Consensus : Oui
Priorisation : Oui
Vue But
But : modéliser, évaluer, sélectionner, configurer
Vue Méthode
Méthode de modélisation : formelle
Méthode d'évaluation : {qualitative, quantitative}
Stratégie d'évaluation : {globale, locale}
Méthode de sélection : MCDM
Contrainte de dépendance : Oui
Vue Outil
Outil de modélisation: langage orienté but
Outil de sélection : Technique (TOPSIS), outil (TOPSiQC)

Cadre 9. Positionnement de la solution proposée

Comme le montre le Cadre 9, notre solution traite la qualité de service dans une perspective intentionnelle. La qualité du service intentionnel est ainsi modélisée par le modèle *MiS-q* qui est interconnecté au modèle *MoS* (Modèle opérationnel de Services) dans l'architecture iSOA. Ce dernier définit l'opérationnalisation des services intentionnels en services logiciels. La définition de la qualité au niveau intentionnel permet de se rapprocher de l'utilisateur final sans autant omettre la perspective logicielle des services. Le modèle *MiS-q* prend en compte les contraintes de dépendance entre les services. Les contraintes définissent des restrictions sur la composition des variantes d'un service.

La solution que nous proposons traite le problème de consensus qualité en proposant un référentiel qualité. Un référentiel qualité compréhensible par l'utilisateur final et connecté aux mesures logicielles. Le référentiel qualité est utilisé par les fournisseurs et les clients du service.

Notre solution gère également les qualités conflictuelles en permettant à l'utilisateur final, via le contexte qualité, de préciser ses préférences en termes de qualité.

Nous proposons une modélisation de la qualité de service via le modèle *MiS-q*, lequel est exprimé, à l'aide du référentiel qualité et de la carte des besoins, dans un langage compréhensible par les utilisateurs finaux et de ses exigences fonctionnelles et non-fonctionnelles. Nous proposons, dans cette thèse, les éléments permettant d'évaluer qualitativement la QoiS globale du service, ainsi que les éléments permettant d'effectuer une évaluation quantitative locale des services logiciels, via les métriques.

Notre solution supporte une sélection à l'exécution des services intentionnels, en fonction des exigences non-fonctionnelles des utilisateurs. Nous proposons également de sélectionner les configurations exécutables du service agrégat, afin de répondre au mieux aux utilisateurs. La sélection et la configuration des services est effectuée en utilisant une méthode d'aide à la décision multicritères, dite TOPSIS.

Notre approche, comme Rohleder (Rohleder,2009) place la qualité au niveau intentionnel pour modéliser, évaluer et configurer. Elle utilise la qualité pour configurer des lignes de produit, formalisée selon le modèle QVaM. Ce modèle s'inspire des travaux de (Bennasri,2005) qui exprime la variabilité du logiciel à un niveau intentionnel, via le modèle de carte. L'originalité de ces travaux est de définir un conteneur logiciel à partir du but utilisateur qu'il permet de satisfaire, et l'agrégation de ces conteneurs est dirigée par les buts grâce aux opérateurs ($\otimes$,*v*, $\cup$,•). Notre proposition *MiS-q* adopte la même vision orientée but, car le modèle de service *MiS* de (Kaabi,2007) s'inspire des travaux de Bennasri qui sont intégrés à l'architecture orientée service.

Notre approche est orientée vers l'utilisateur final, et elle permet d'établir un consensus qualité entre les différentes parties prenantes du processus de sélection et de configuration, à savoir les fournisseurs et les utilisateurs finaux d'un même domaine. Il s'agit d'uniformiser le vocabulaire (les qualités et leurs évaluations) de façon à ce que l'utilisateur final les comprend, et le fournisseur les exploitent pour décrire la qualité de leurs services intentionnels et logiciels, en rapprochant les mesures quantitatives délivrées par les services logiciels aux évaluations qualitative des services de haut niveau. Aussi, ce vocabulaire est partagé entre les fournisseurs d'un même domaine permettant ainsi d'améliorer leur crédibilité. Rohleder, quant à elle, utilise le cadre NFR pour modéliser la qualité de service dans le cas de configuration d'une ligne de produit.

La solution que nous proposons permet de sélectionner dans un annuaire, des services réutilisables et autonomes, et de configurer un service agrégat qui est assimilé à une ligne de

produit. Rohleder fait de la configuration d'une ligne de produit donnée. Nous adoptons la méthode multicritères TOPSIS pour comparer les services et les configurations de service. Notre choix est justifié par le fait que TOPSIS fournit une solution euclidienne, laquelle est simple à mettre en œuvre et facile à comprendre par l'utilisateur final. En effet, ce dernier peut comprendre plus naturellement la déviation d'un service ou d'une configuration par rapport à ce qu'il souhaite obtenir. Rohleder utilise les mesures de similarité, définies par (Zoukar,2005), pour comparer les concepts du profil du produit à ceux du modèle QVaM. Ces mesures sont complexes et elles ne sont pas faciles à comprendre, car les résultats obtenus sont difficiles à interpréter par l'utilisateur final. De plus, le calcul de similarité se fait localement, en comparant les concepts, alors que notre approche possède une vision plus globale. En effet, TOPSIS permet de comparer les services et les configurations, en calculant leurs qualités globales. De plus, dans le cas de configuration, les approches de ligne de produit (Djebbi,2011) ont démontré que les contraintes de dépendance entre variantes doivent être prises en compte afin de dériver des configurations cohérentes. C'est pour cette raison que notre approche propose de considérer les contraintes de dépendance pour dériver des configurations exécutables du service agrégat. Rohleder, quant elle, ne considère pas les contraintes de dépendance qui permettent de valider les configurations du modèle QVaM.

Chacun des éléments de notre proposition sera détaillé au fur et à mesure dans les différents chapitres de cette thèse. Le chapitre 4 présente l'architecture intentionnelle des services (*iSOA-intentional Service Oriented Architecture*), tandis que le chapitre 5 présente le méta-modèle du référentiel qualité. Le chapitre 6 et chapitre 7 décrivent respectivement le méta-modèle intentionnel de la qualité de service (*MiS-q*) et la méthodologie permettant la publication du modèle *MiS-q*. Le chapitre 8 présente le processus de sélection intentionnelle, lequel présente également le contexte qualité et la configuration intentionnelle des services. Le chapitre 9 décrit le prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*). Le chapitre 10 illustre l'approche proposée par un exemple, tandis que le chapitre 11 conclut ce mémoire de thèse en résumant la plus-value de ce travail et en proposant des perspectives de recherche.

CHAPITRE 4 : ARCHITECTURE ORIENTEE SERVICE INTENTIONNEL (iSOA)

4.1 *INTRODUCTION*

Dans l'approche SOC, les services sont décrits par des interfaces qui ne sont pas forcément compréhensibles par les utilisateurs métier, car il existe un problème de discordance conceptuelle entre les exigences fonctionnelles formulées par ces utilisateurs et les descriptions des services logiciels (formulés bien souvent en WSDL). Kaabi (Kaabi,2007) suggère de déplacer la vision technique du service vers une vision orientée-intention, à travers l'architecture orientée service intentionnel (*iSOA-intentional Service Oriented Architecture*).

Nous adoptons l'architecture iSOA dans cette thèse afin de diminuer la discordance conceptuelle entre l'expression des exigences fonctionnelles et la description des services logiciels. L'architecture iSOA propose la définition d'un modèle de représentation des services dénommé *MiS* (Modèle intentionnel de Services). Il s'agit, en fait, d'un modèle qui permet de représenter, à un niveau intentionnel, les services du système à développer. Ces services, dénommés *services intentionnels*, permettent d'aider un client à exprimer son intention et se concentrer sur les buts qu'un service permet de réaliser, plutôt que sur ses fonctionnalités. Nous adoptons particulièrement l'architecture iSOA, car d'une part le modèle *MiS* est exprimé dans un langage compréhensible par les utilisateurs, et d'autre part le service modèle *MiS* est interconnecté au modèle *MoS* (Modèle opérationnel de Services), lequel définit l'opérationnalisation des services intentionnels en services logiciels. Le modèle de la Carte (Rolland&Prakash,2001) est utilisé pour représenter les exigences fonctionnelles sous forme de buts métier. Les services intentionnels sont générés à partir de la carte représentant les besoins des agents métier (Kaabi,2007).

Nous proposons dans cette thèse de bénéficier des avantages de l'architecture iSOA afin de résoudre le problème de discordance conceptuelle, lequel est toujours persistant, entre l'expression des exigences non-fonctionnelles et la description de la qualité des services logiciels. Nous définissons alors une extension du modèle *MiS* afin de considérer la dimension qualitative du service intentionnel, ainsi que de la méthodologie de génération des services.

Ce chapitre est composé de quatre sections permettant de rappeler les éléments, que nous réutilisons, de l'architecture iSOA développée par Kaabi (Kaabi,2007). La définition de l'architecture iSOA est présentée dans la section 2. Les concepts du méta-modèle *MiS* sont développés dans la section 3. Ensuite, le modèle de la Carte est présenté dans la section 4, tandis que le processus permettant de générer les différents types de services du modèle *MiS* est développé dans la section 5. Finalement, la section 6 conclut ce chapitre.

4.2 L'ARCHITECTURE ORIENTEE SERVICE INTENTIONNEL

L''architecture orientée service intentionnel (*iSOA-intentional Service Oriented Architecture*), proposée par Kaabi, Rolland et leurs collègues (Kaabi,2007) (Rolland et *al.*,2007), est présentée à la Figure 14. Dans l'architecture iSOA, les services sont décrits de façon intentionnelle : en termes de buts et de stratégies permettant d'atteindre ces buts, leur publication, leur recherche et leur composition se fait sur la base de ces descriptions intentionnelles (Kaabi et *al.*,2004) (Rolland&Kaabi,2007) (Rolland et *al.*,2008).

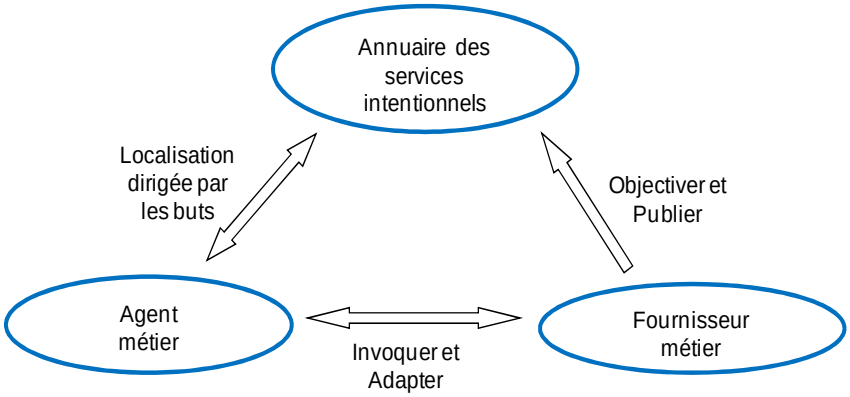

Figure 14. Le modèle iSOA

L'architecture iSOA se distingue de SOA par les deux points suivants :

(1) les agents métiers remplacent les agents logiciels au niveau des interactions ; et

(2) la description intentionnelle du service remplace la description technique.

Dans la vision iSOA, les organisations décrivent les services dans un mode intentionnel (buts et stratégies) et les publie dans un annuaire de services intentionnels. L'agent métier effectue une découverte, dirigée par les buts, d'un ou plusieurs services correspondant à ses exigences fonctionnelles. Une fois le service localisé, l'agent métier peut interagir directement avec le fournisseur métier et invoquer les services.

L'architecture iSOA se base sur deux niveaux d'abstraction : le *niveau opérationnel* décrivant les services logiciels et le *niveau intentionnel* décrivant les services intentionnels. La Figure 15 montre la correspondance entre l'agent métier et le service logiciel : le service intentionnel correspond aux exigences fonctionnelles des agents métier et il abstrait les fonctionnalités du service logiciel.

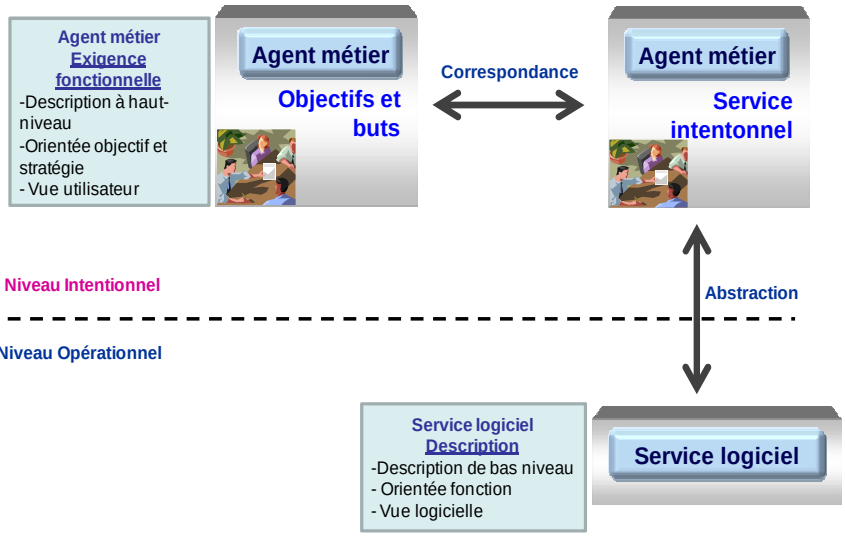

Figure 15. Correspondance entre l'agent métier et le service logiciel, via le service intentionnel

L'analyse de l'architecture iSOA nous a permis de définir les caractéristiques suivantes :

- elle abstrait les détails du service technique et de ses fonctionnalités, grâce au concept de service intentionnel. Le service intentionnel correspond aux exigences fonctionnelles des agents métier : il se concentre sur l'essence du service, à savoir le but que le service permet d'atteindre ;
- elle gère la variabilité dans la manière d'atteindre le but associé au service intentionnel en définissant les différentes manières d'atteindre le but ;
- elle exprime la composition des services intentionnels des clients ;
- elle constitue une démarche permettant d'identifier et de décrire les services du métier qui sont publiés dans l'annuaire des services en utilisant un graphe de buts et de stratégies (modèle de la Carte) ; et
- elle supporte l'opérationnalisation du service intentionnel vers les services logiciels.

Dans l'architecture iSOA, le modèle intentionnel de service (*MiS*) est défini pour décrire le service intentionnel, lequel peut être opérationnalisé par un service logiciel. Le méta-modèle *MiS* est présenté dans la section suivante.

4.3 *LE MODELE INTENTIONNEL DE SERVICE (MiS)*

La Figure 16 présente le Modèle intentionnel de Service, nommé *MiS*, en utilisant le langage UML (*Unified Modeling Language*). Le modèle *MiS* est composé de trois principaux aspects permettant de décrire un service intentionnel, à savoir l'interface, le comportement et la composition. Nous détaillons, dans cette section, ces trois aspects, lesquels sont relatifs aux différents concepts du modèle *MiS*. Nous proposons d'illustrer les concepts de la Figure 16 par l'exemple d'approvisionnement de produits présenté dans (Rolland&Prakash,2000), et repris par Kaabi (Kaabi,2007) pour l'identification des services.

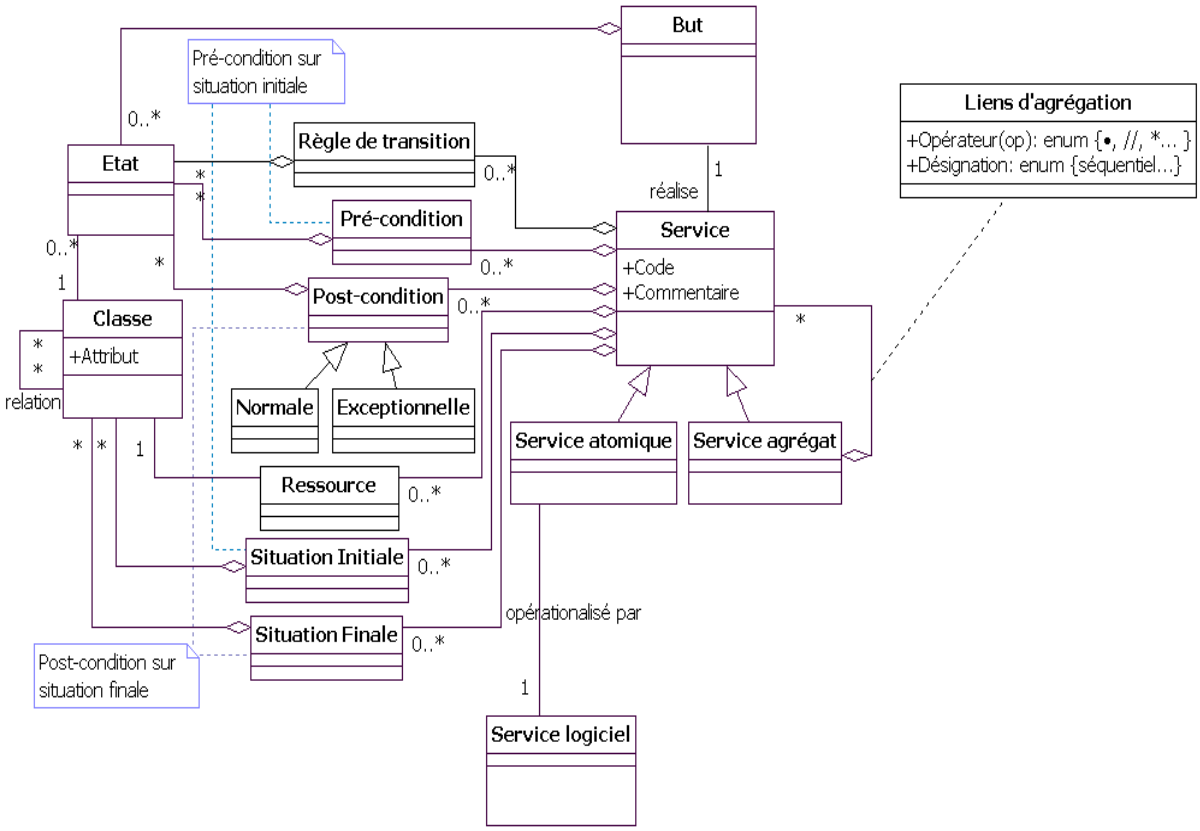

Figure 16. Le modèle *MiS*

4.3.1 L'INTERFACE DU SERVICE INTENTIONNEL

Selon Kaabi (Kaabi,2007), le premier aspect descriptif d'un service intentionnel est l'*interface*. L'interface d'un service intentionnel permet d'exposer les caractéristiques principales du service aux clients potentiels et de permettre sa réutilisation dans un processus de composition de services. Dans le modèle *MiS*, l'interface a la particularité d'associer à chaque service la connaissance intentionnelle (*i.e.* le but de la Figure 16) et situationnelle (*i.e.* les situations initiale et finale de la Figure 16) permettant la réalisation du but associé au

service.

Le service intentionnel est identifié par un code. Un service réalisant le but *a* est désigné dans (Kaabi,2007) par le code $\boldsymbol{S}_a$ (***S*** pour service et *a* pour le but que ce service réalise). Par exemple, le service $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ fait référence à un service permettant de réaliser le but *Effectuer l'approvisionnement en produits*.

Le but qui donne son identité au service intentionnel joue un rôle important lorsque l'agent métier (ou client) souhaite rechercher le service correspondant à son besoin (qui reflète son intention). La description du service par un but permet ainsi de diminuer la discordance conceptuelle entre les exigences fonctionnelles des agents métier et la description des services techniques.

Dans la perspective d'effectuer une recherche efficace, le but du service intentionnel se base sur une approche linguistique initialement développée par (Prat,1997). Cette approche est inspirée de la grammaire des cas de Fillmore (Fillmore,1968) et des extensions de (Dik,1989). Elle se base sur le fait que la sémantique d'un but est capturée par un verbe et un ou plusieurs paramètres. Le but du service, comme le montre la Figure 17, comprend obligatoirement un *verbe* et une *cible* et optionnellement un ensemble de *paramètres*.

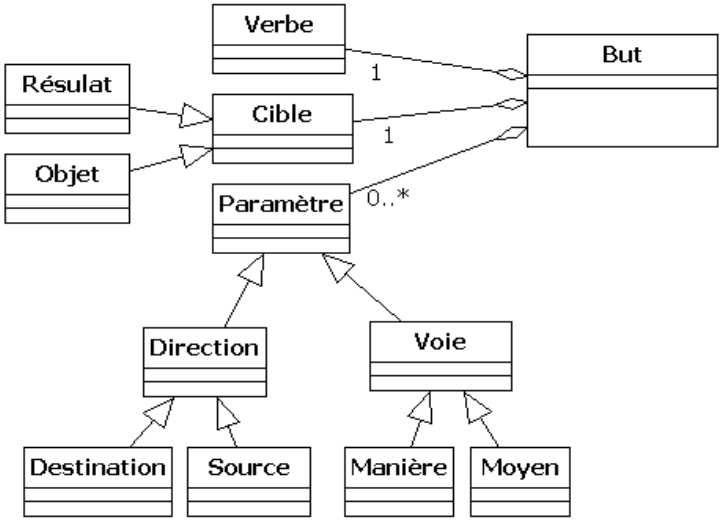

Figure 17. Le modèle de formalisation du but du service

La classe du verbe détermine son sens. La cible est le complément d'objet du verbe, elle désigne la ou les entité(s) impactée (s) par le but. Par exemple, $(Effectuer)_{verbe}$ $(l'approvisionnement\ en\ produits)_{cible}$. Plusieurs paramètres peuvent être associés à un verbe.

La cible peut être l'*objet* ou le *résultat*. L'objet correspond à l'entité qui existe avant la réalisation du but, alors que le résultat correspond à l'entité qui découle de la réalisation du but. Par exemple, dans le cas du but $(Effectuer)_{verbe}$ $(une\ commande)_{résultat}$, la commande

résulte de la réalisation du but. Pour le but *(Confirmer)*$_{verbe}$ *(une commande)*$_{objet}$, la commande existe avant la réalisation du but.

Le paramètre de *direction* caractérise la *source* et la *destination* des cibles liées au but. Par exemple, le but *(Effectuer)*$_{verbe}$ *(une commande)*$_{résultat}$*(à partir d'une demande)*$_{source}$ comprend le verbe *Effectuer*, la cible *une commande* et la source *à partir d'une demande*, c'est-à-dire que la commande est effectuée à partir de la demande. Dans le cas du but *(Stocker)*$_{verbe}$ *(une commande)*$_{objet}$ *(dans la base de données)*$_{destination}$, la destination (*i.e. dans la base de données*) précise l'emplacement de la cible (*i.e. commande*) produite par la réalisation du but.

Les paramètres de *voie* font référence aux instruments et aux approches de réalisation d'un but. Le *moyen* décrit l'outil permettant d'atteindre le but. Dans le but *(Payer)*$_{verbe}$ *(une facture)*$_{objet}$*(par carte bancaire)*$_{moyen}$, la carte bleue est le moyen choisi pour réaliser le but *Payer une facture*. La *manière* décrit l'approche (ou la façon) permettant de réaliser le but. Dans le but *(Annuler)*$_{verbe}$ *(une commande)*$_{résultat}$ *(par expiration du délai d'attente)*$_{manière}$, l'approche permettant de réaliser le but *Annuler une réservation* est « *expiration du délai d'attente* ».

La formulation linguistique du but est complétée par la définition d'un ensemble d'états désirés (le lien entre but et état de la Figure 16 traduit cet aspect). Les états sont ceux des classes d'objets de gestion considérées dans le système. La notation pointée usuelle *NomClasse.NomAttribut = 'valeur'* est utilisée pour formuler les états. Par exemple, le but *Annuler Réservation* correspond à l'état *réservation.statut = 'annulée'* où *réservation* est le nom de la classe d'objets et *statut* est un attribut caractéristique de la réservation.

La situation initiale définit la situation de départ dans laquelle le service peut être exécuté pour satisfaire le but du service. La situation finale définit la situation d'arrivée une fois le service exécuté. La description de la situation initiale et finale définissent respectivement les paramètres d'entrée et de sortie du service. Dans les deux cas, et comme le montre la Figure 16, les situations sont modélisées comme des classes d'objets (Kaabi,2007). Par exemple, le service $\mathbf{S}_{Annuler\ réservation}$ permet de réaliser le but *Annuler réservation*, sa situation initiale est identique à sa situation finale, laquelle correspond à la classe *réservation*.

L'interface d'un service intentionnel a la même sémantique que celle de l'interface d'un service logiciel. Elle permet de décrire le service d'un point de vue externe : c'est la partie visible du service, laquelle est utilisée d'une part pour construire des compositions de service, et d'autre part pour localiser les services correspondant aux exigences fonctionnelles des clients. L'interface du service intentionnel comporte le but, la situation initiale qui définit les

paramètres d'entrée et la situation finale qui fournit les paramètres de sortie (voir (Kaabi,2007) pour plus de détail).

4.3.2 LE COMPORTEMENT DU SERVICE INTENTIONNEL

D'après Kaabi (Kaabi,2007), le deuxième aspect descriptif du service intentionnel est le *comportement*. Le comportement d'un service intentionnel correspond aux pré/post-conditions (*i.e.* éléments Pré-condition et Post-condition de la Figure 16) qui sont des conditions sur les états des classes d'objets des situations initiale et finale, respectivement.

Selon le modèle *MiS*, un service est vu comme la transition d'une pré-condition vers une post-condition, résultant de la réalisation du but du service, par l'exécution de règles de transition associées à ce service et spécifiées par l'élément Règle de transition (*cf.* Figure 16). Celles-ci permettent de préciser les contraintes régissant la réalisation des opérations entraînant le changement d'états des classes d'objets.

Les pré-conditions et les post-conditions font référence à des états de classes. La pré-condition permet de préciser les états des classes constituant les conditions devant être respectées pour que le but du service puisse être réalisé. La post-condition permet de préciser les états des classes obtenus suite à la réalisation du but du service. Les post-conditions peuvent être de deux types : normales ou exceptionnelles (éléments Normale et Exceptionnelle de la Figure 16). La post-condition est normale dans le cas où le service s'exécute normalement. La post-condition est exceptionnelle lorsque le service ne peut pas s'exécuter, dans le cas par exemple de pré-conditions non vérifiées (voir (Kaabi,2007) pour plus de détail).

4.3.3 LA COMPOSITION DU SERVICE INTENTIONNEL

Selon Kaabi (Kaabi,2007), le troisième aspect descriptif du service intentionnel est la *composition de services*. La composition de service permet de fournir des services de grosses granularités. Comme le montre la Figure 16, il y a différents types de services intentionnels. Les services peuvent être des services *atomiques* ou des services *agrégats*. Un service atomique réalise un but opérationnalisable et il est associé à une orchestration de services logiciels (Kaabi,2007). A l'opposé, un service agrégat réalise un but tactique ou stratégique et sa composition reproduit le type d'affinement *Et*/*Ou* du but du service. Ainsi, le service agrégat peut être un *service composite* ou un *service à variation*. Un service composite est justifié par un affinement *Et*, tandis qu'un service à variation correspond à un affinement *Ou* du but du service.

Nous détaillons chacun de ces services dans la suite de cette section.

4.3.3.1 *LE SERVICE ATOMIQUE*

Comme défini dans Kaabi (Kaabi,2007), le service atomique est celui qui ne peut être composé d'autres services intentionnels. L'atomicité d'un service intentionnel est justifiée par la perspective intentionnelle : il hérite de la nature du but qui le caractérise, c'est un service dont le but est directement opérationnalisable, sans avoir besoin d'être affiné en sous-buts. Le service atomique a vocation d'être réutilisé comme brique de base dans la construction de services intentionnels de granularité plus importante

Comme le montre la Figure 16, l'opérationnalisation du service intentionnel atomique peut être assurée par l'exécution d'un service logiciel ou d'une composition de services logiciels. le concept de service logiciel est défini dans l'approche iSOA par le modèle *MoS* (Kaabi,2007). Kaabi (Kaabi,2007) précise qu'il existe une relation bidirectionnelle entre le modèle *MiS* et le modèle *MoS* : (i) le service intentionnel atomique du modèle *MiS* est opérationnalisé par l'exécution d'un service logiciel du modèle *MoS* ; et (ii) le service logiciel délivre un résultat lequel est considéré comme l'information nécessaire pour que le but du service atomique soit réalisé. Cette relation bidirectionnelle des modèles *MiS* et *MoS* permet de faire la transition entre la notion de service à un niveau intentionnel et celle à un niveau opérationnel. Ceci met en avant le fait que les deux notions de service coexistent et coopèrent à la fois au niveau intentionnel et au niveau opérationnel.

Notre intérêt dans cette thèse se porte particulièrement sur l'extension du modèle *MiS* afin de prendre en compte l'opérationnalisation du service atomique par plusieurs services logiciels, lesquels fournissent diverses qualités permettant de répondre, de différentes manières, aux exigences non-fonctionnelles des clients. Néanmoins, il n'est pas nécessaire de détailler les concepts du modèle *MoS* ainsi que la méthodologie d'opérationnalisation, car ce dernier est hors sujet de cette thèse (pour plus de détail voir (Kaabi,2007)).

La Figure 18 illustre la représentation graphique d'un service atomique, laquelle correspond à un rectangle contenant le code du service. Celui-ci est composé habituellement de la lettre ***S*** (référençant le service) et d'un indice exprimant le but que le service réalise. Dans l'exemple de l'approvisionnement de produits, le but *Contrôler le stock en continu* peut être associé au service atomique $\boldsymbol{S}_{Contrôler\ le\ stock\ en\ continu}$, graphiquement représenté à la Figure 19.

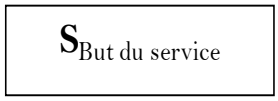

Figure 18. Représentation graphique du service atomique (Kaabi,2007)

Figure 19. Représentation graphique de $\mathbf{S}_{\text{Contrôler le stock en continu}}$

4.3.3.2 *LE SERVICE AGREGAT*

Contrairement au service atomique, le service agrégat cherche à réaliser un but tactique ou stratégique (Kaabi,2007). Il est formé par l'agrégation de plusieurs services intentionnels (atomiques ou agrégats). Cette agrégation est déterminée à partir des liens d'affinement *Et*/*Ou* du but du service. Ces liens d'affinement définissent deux types de service agrégat : les services composites et les services à variation. Les premiers (*i.e.* les services composites) sont reliés par un lien *Et*, tandis que les derniers (*i.e.* les services à variation) correspondent à un lien d'affinement *Ou*. Autrement dit, la nature du service agrégat dépend du lien existant entre les services composants. Lorsque ces derniers sont reliés par un lien *Et*, le service agrégat est composite. Le service agrégat à variation correspond à un lien *Ou* entre les services composants. On dit alors qu'un service agrégat se décompose jusqu'à obtenir des services atomiques exécutables. Le service agrégat est appelé aussi « service navigationnel » (Kaabi,2007), car il est exécuté par navigation dans l'arbre d'affinement *Et/Ou*. Ainsi, la composition d'un service intentionnel est elle-même orientée but. Elle est représentée dans le modèle *MiS* par des liens d'agrégation (*cf.* Figure 16), lesquels expriment par des opérateurs spécifiques (« $\bullet$ », « // », « * », « $\otimes$ », « $\vee$ » et « $\cup$ ») les relations de type *Ou Exclusif*, *Ou* et *Et* entre les services intentionnels.

4.3.3.2.1 LE SERVICE COMPOSITE

Un service composite reflète un lien de précédence/succession entre les buts qu'il réalise. Il s'appuie sur un affinement de type *Et* du but du service en sous-buts. Trois types de services composites existent : le service séquentiel, le service parallèle et le service itératif.

Le service séquentiel est défini lorsque qu'un ordre dans la livraison des services composants est nécessaire. Le service composite séquentiel est de même niveau d'abstraction que ceux qui le composent, mais de niveau de granularité intentionnelle plus élevé. L'opérateur de composition « $\bullet$ » est associé au service composite séquentiel. Celui-ci est noté : $\boldsymbol{S}_s = \bullet(\boldsymbol{S}_1, \boldsymbol{S}_2 \ldots \boldsymbol{S}_n)$, tel que *n* est le nombre des services composants.

Le service parallèle peut être exécuté (ou réalisé) selon un ordre quelconque et non dans un ordre impératif, comme dans le cas précédent (celui des services composites séquentiels). L'opérateur de composition « // » est associé au service composite parallèle, lequel est noté : $\boldsymbol{S}_p = //(\boldsymbol{S}_1, \boldsymbol{S}_2 \ldots \boldsymbol{S}_n)$, tel que *n* est le nombre des services composants.

Le service itératif est défini lorsqu'un service intentionnel peut être exécuté récursivement. L'itération peut s'appliquer aussi bien à un service atomique entrant dans une composition séquentielle ou parallèle, ou à un service agrégat dans son ensemble (*i.e.* itération dans la composition). L'opérateur de composition « * » est associé au service composite itératif. Par exemple, dans le service composite séquentiel $S_s = \bullet (S^*_1, S_2 \ldots S_n)$, le service S_1 est le service itératif.

La Figure 20 illustre la représentation graphique d'un service composite. Elle met en évidence les trois opérateurs présentés précédemment (*i.e.* « • », « // », « * »). A la Figure 20 (a), nous avons un service composite séquentiel qui fait appel à l'opérateur d'itération concernant l'un de ses services atomiques (S_1). La Figure 20-(b) montre la représentation graphique d'un service composite séquentiel dont le service composant est lui-même un service agrégat (à variation) qui nécessite une exécution itérative. Enfin, la Figure 20-(c) illustre un service parallèle (opérateur de composition parallèle « // »).

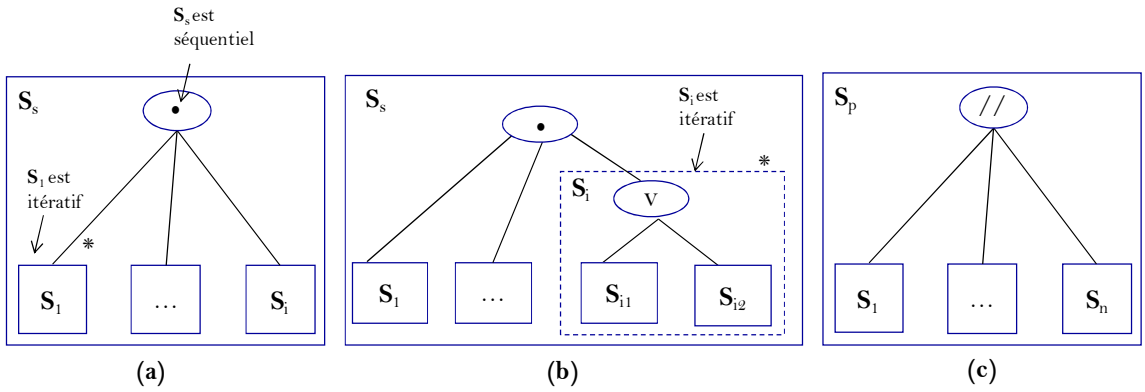

Figure 20. Représentation graphique du service composite

La Figure 21 montre la représentation graphique du service composite séquentiel $S_{Entrer\ les\ produits\ en\ stock\ normalement}$. Ce service nécessite la réalisation du but *Entrer en stock les produits livrés* qui ne peut se faire que si l'acquisition des produits est préalablement effectuée, c'est-à-dire que si le but *Acquérir des produits* est réalisé. Ceci se traduit par une composition séquentielle de services : $S_{Entrer\ les\ produits\ en\ stock\ normalement} = \bullet(S_{Acquérir\ des\ produits}, S_{Entrer\ en\ stock\ les\ produits\ livrés})$.

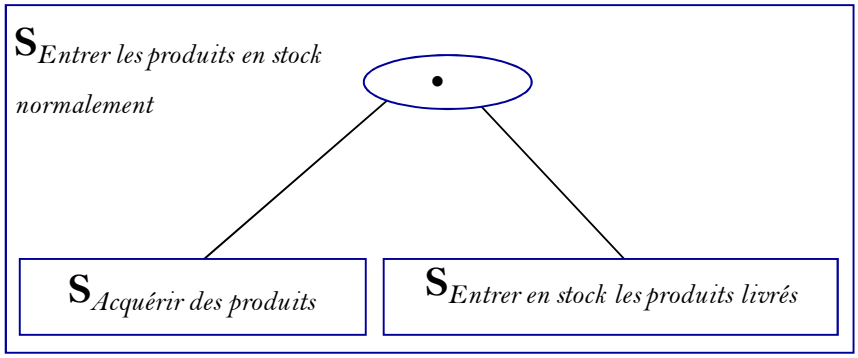

Figure 21. Représentation graphique du service $S_{Confirmer\ réservation}$

4.3.3.2.2 LE SERVICE A VARIATION

Un service à variation introduit de la variabilité dans la modélisation du service intentionnel. Il s'appuie sur un affinement de type *Ou* du but du service en sous-buts, représentant ainsi les différentes alternatives de réalisation du but principal. Trois types de services à variation existent : le service à choix alternatif, le service à choix multiple et le service multi-chemin.

Le service à choix alternatif exprime un choix parmi plusieurs services composants, lesquels sont mutuellement exclusifs, ce qui correspond à un affinement de type *Ou exclusif*. Au moment de l'exécution, un seul service est choisi parmi l'ensemble des alternatives offertes. L'opérateur de variation « ⊗ » est associé au service à choix alternatif, noté : $S_a = \otimes (S_1, S_2 \dots S_n)$, où *n* représente le nombre des services composants.

Le service à choix multiple exprime un choix non-exclusif entre les services composants, ce qui correspond à un affinement *Ou*. Il regroupe plusieurs services composants parmi lesquels au moins un, mais éventuellement plusieurs, peuvent être choisis. Le choix multiple est associé à l'opérateur de variation « *v* ». Ainsi, un service à choix multiple S_m est noté : $S_m = v (S_1, S_2 \dots S_n)$, où *n* correspond au nombre des services composants.

Dernier type de service à variation, le service multi-chemin introduit une variation qui porte sur un « chemin » de buts, c'est-à-dire sur des enchaînements alternatifs de buts. Il regroupe plusieurs chemins parmi lesquels un seul peut être choisi. L'opérateur de variation « ∪ » est associé au service à choix multiple, lequel est noté : $S_m = \cup (S_1, S_2 \dots S_n)$, tel que *n* est le nombre des services composants.

La Figure 22 illustre la représentation graphique d'un service à variation. Elle met en évidence la représentation d'un service à choix alternatif (*cf.* Figure 22-(a)), d'un service à choix multiple (*cf.* Figure 22-(b)) et d'un service multi-chemin (*cf.* Figure 22-(c)).

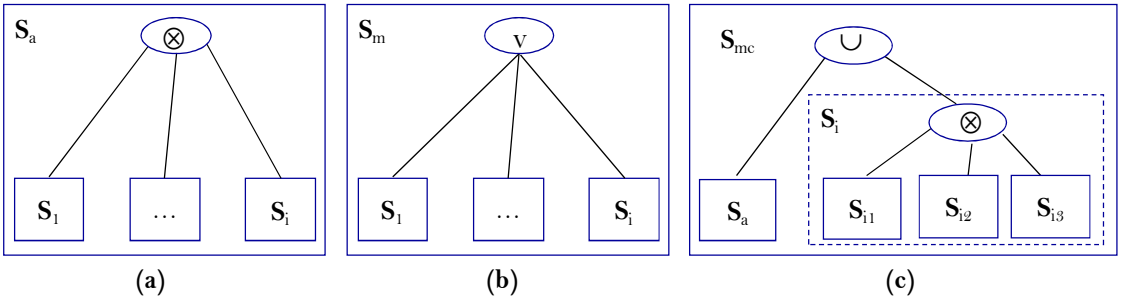

Figure 22. Représentation graphique du service à variation

Dans l'exemple de l'approvisionnement de produits, la Figure 23 montre la représentation graphique du service intentionnel réalisant le but *Acquérir des produits par planification*. Celle-ci peut se faire soit par la réalisation du but *Commander des produits par seuil de réapprovisionnement*, soit par le but *Commander des produits par prévision stratégique*. Ceci se traduit par le service à variation suivant : $\mathbf{S}_{Acquérir\ des\ produits\ par\ planification} = \otimes (\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}, \mathbf{S}_{Commander\ des\ produits\ par\ prévision\ stratégique})$.

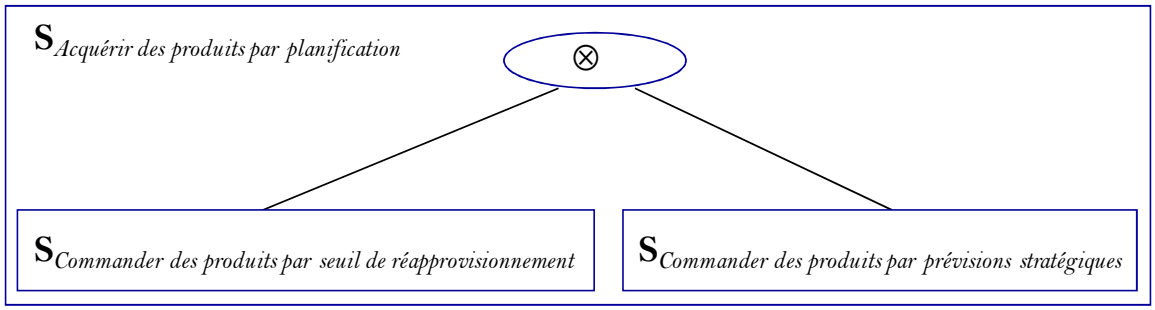

Figure 23. Représentation graphique du service $\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$

Dans l'approche iSOA, les services intentionnels publiés dans l'annuaire de services sont issus des processus métier des organisations. Ces services sont en relation avec les besoins métier et, manifestement, aident à les atteindre. Il est alors important de donner au fournisseur métier les moyens de construire un modèle du métier à partir duquel il est possible de générer les services, de les décrire selon les termes de *MiS* et de les publier. Kaabi (Kaabi,2007) a choisi d'utiliser le modèle de la Carte (Rolland&Prakash,2000) pour modéliser le métier des organisations dans des termes intentionnels et de développer un ensemble de règles méthodologiques pour dériver les services d'une carte intentionnelle.

La prochaine section introduit le modèle de la Carte pour la capture des besoins, suivi du processus de génération des services à partir de la carte.

4.4 *LE MODELE DE LA CARTE*

Les besoins des utilisateurs sont décrits avec un modèle orienté but appelé la Carte (Rolland,1999). La *Carte* est un système de représentation des processus dans un mode intentionnel. Le système de représentation de la carte utilise le concept de *but* et se différencie des autres modèles par l'introduction du concept de *stratégie*, lequel permet de définir la manière de réaliser un but. Les cartes proposent un mécanisme d'affinement qui peut être modélisé par un lien d'affinement allant d'un couple but/stratégie vers une autre carte. L'affinement permet de décrire les buts et stratégies à différents niveaux de détail.

Selon Kaabi (Kaabi,2007), le modèle de la Carte est adapté à la découverte des services. Il met en évidence les différentes stratégies pour satisfaire le même but et aussi les différentes combinaisons de stratégies et de buts pour atteindre un but final. Particulièrement, il supporte la découverte de la variabilité des services. Chaque combinaison possible de buts et de stratégies identifie une variante de services possible pour atteindre le but du service global.

D'après Rolland et Prakash (Rolland&Prakash,2001) (Rolland,2007), une carte est un modèle de processus dans lequel un ordonnancement non déterministe de buts et de moyens (stratégies) de réaliser ces buts est représenté. Le méta-modèle présenté à la Figure 24 montre que :

- Une *carte* a un *code* et un *libellé* qui correspond au but de la carte.
- Une *carte* est composée d'au moins deux, et éventuellement de plusieurs sections. Une *section* est une agrégation d'un but source, d'un but cible et d'une stratégie. Chaque section correspond à une stratégie pouvant être utilisée pour réaliser un but cible, une fois que le but source est atteint.

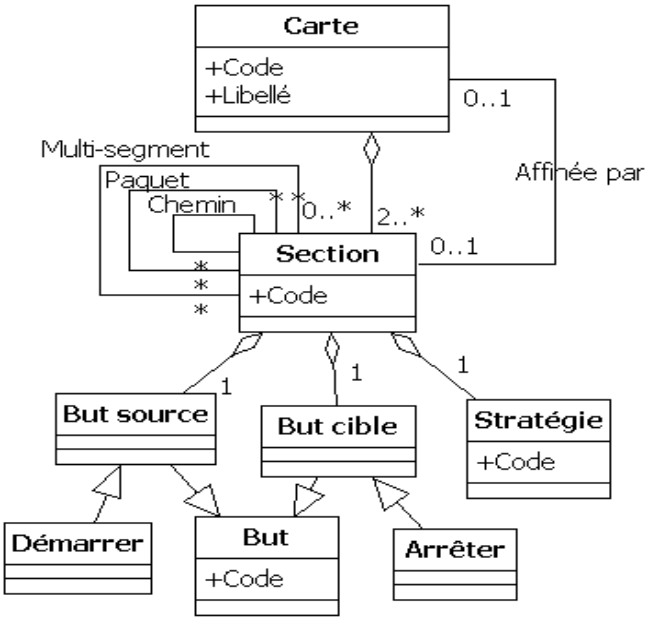

Figure 24. Le méta-modèle de la Carte (Rolland&Prakash,2000)

Graphiquement, une *carte* est un graphe orienté (*cf.* Figure 25) dans lequel les nœuds sont des buts et les arcs correspondent aux stratégies. Un arc entrant identifie une *stratégie* qui permet de réaliser le but associée au nœud cible. Les cartes proposées dans l'architecture iSOA s'intéressent à la modélisation des services intentionnels, en faisant une jonction du niveau stratégique (*i.e.* les services agrégats) et opérationnel (*i.e.* les services atomiques).

La Figure 25 présente un exemple de carte de gestion de stock (Rolland&Prakash,2000) (Kaabi,2007) qui a comme but *Effectuer l'approvisionnement en produits*. Dans ce qui suit, nous définissons les différents éléments utilisés dans le méta-modèle de la Carte. L'exemple de la Figure 25 comporte quatre buts (*Démarrer*, *Acquérir des produits*, *Contrôler le stock* et *Arrêter*), douze stratégies (*Par seuil de réapprovisionnement*, *Par prévisions stratégiques*, *Manuellement*, *Entrer en stock des produits livrés*, *Entrer les produits directement*, *Périodique*, *Echantillonnage*, *En continu*, *Par valuation*, *Par inspection qualité*, *Par transfert des produits*, *Par contrôle du paiement des factures*) et douze sections.

Pour faciliter la manipulation des cartes, les buts, les stratégies et les sections sont codés. il existe une codification locale et une codification absolue (Rolland&Prakash,2001). La codification locale consiste à affecter d'abord une lettre de l'alphabet à chaque but de la carte. Par exemple, les buts *Démarrer*, *Acquérir des produits*, *Contrôler le stock* et *Arrêter* de la Figure 25 sont respectivement codés par *a*, *b*, *c* et *d*. Ensuite, les stratégies sont codées par des nombres relatifs à leurs buts cible et source. Ainsi, deux stratégies ayant les mêmes buts source et cible seront respectivement codées par deux nombres distincts et consécutifs. Les

stratégies de l'exemple de la Figure 25 sont codées comme suit : pour le but source *a* et le but cible *b* les stratégies sont numérotées *1*, *2* et *3*, pour le but source *b* et le but cible *c* il y a une stratégie et elle numérotée *1*, pour le but source *c* et le but cible *c* les stratégies sont numérotées *1*, *2*, *3*, *4*, *5* et 6, *etc*. enfin, les sections sont codées par juxtaposition des trois éléments (i) la lettre du but source, (ii) la lettre du but cible et (iii) le nombre de la stratégie. Par exemple, la section ab_1 permet, en partant du but *a*, d'atteindre le but *b* en suivant la stratégie *1*. La codification absolue, quant à elle, consiste à coder la carte racine par C. Ses buts sont nommés C.a, C.b, C.c, *etc*. et ses sections sont codifiées en $C.ab_1$, $C.ab_2$, $C.ab_3$, *etc*.

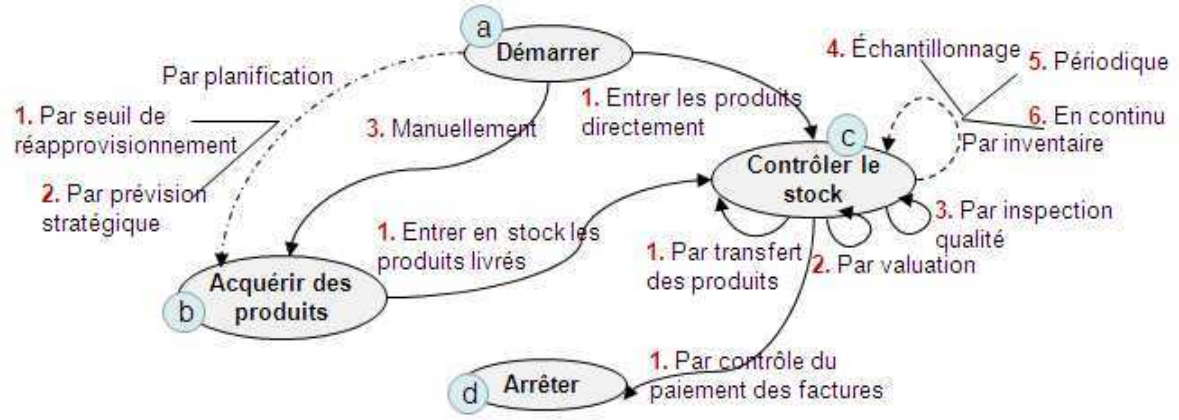

Figure 25. La carte du but *Effectuer l'approvisionnement en produits* (Kaabi,2007)

4.4.1 LE BUT

Un *but* est l'objectif qu'on veut atteindre. Jackson (Jackson,1995) définit le but comme étant une déclaration « optative » qui exprime ce qu'on veut, un état ou un résultat qu'on souhaite atteindre. Par exemple, comme le montre la Figure 25, *Acquérir des produits* et *Contrôler le stock* sont deux buts dans le domaine de la gestion de stock.

Comme le montre le méta-modèle de la Carte (*cf.* Figure 24), chaque carte possède deux buts particuliers : *Démarrer* et *Arrêter*. Le but Démarrer permet de commencer l'exécution d'un processus et le but Arrêter permet de le terminer. Le but *Démarrer* est un but source et le but *Arrêter* un but cible.

Dans l'architecture iSOA (Kaabi,2007), le but capture "l'essence" du processus, à savoir l'objectif métier qu'il permet d'atteindre, en faisant abstraction des caractéristiques techniques de sa réalisation.

4.4.2 LA STRATEGIE

Une *stratégie* (Rolland&Prakash,2001) est une approche, une manière ou un moyen de réaliser un but cible à partir d'un but source. Dans l'architecture iSOA (Kaabi,2007), les stratégies correspondent aux différentes alternatives permettant de réaliser le but du service.

L'exemple de la Figure 25 montre que la demande d'acquisition des produits peut être faite manuellement ou générée automatiquement à l'aide de stratégies de planification. Les deux stratégies *Manuellement* et *Par planification* sont donc des manières différentes de réaliser le but *Acquérir des produits*.

4.4.3 LA SECTION

Une *section* (Rolland&Prakash,2001) est l'élément clé de la carte. Elle représente le triplet <*but-source, but-cible, stratégie*>. Une section décrit ainsi une manière spécifique (*stratégie*) de réaliser le but cible (*but-cible*) une fois que le but source (*but-source*) est réalisé.

L'exemple de la Figure 25 montre un ensemble de sections dont la section bc1 <*Acquérir des produits, Contrôler le stock, Entrer en stock des produits livrés*>. Celle-ci permet de contrôler les livraisons, une fois les produits livrés, en appliquant la stratégie *Entrer en stock des produits livrés*.

4.4.4 LES RELATIONS ENTRE LES SECTIONS

Rolland et Prakash (Rolland&Prakash,2001) ont défini trois types de relation entre les sections de la carte (*cf.* Figure 24). Le *paquet* correspond à des relations de type *Ou Exclusif* entre les sections. Le *multi-segment* lie les sections par des relations de type *Et/Ou*. Enfin, le *chemin* représente la relation de précédence entre les sections.

Nous présentons ces trois relations dans les sous-sections suivantes.

4.4.4.1 LE PAQUET

La relation de *paquet* entre les sections précise que des sections ayant les mêmes buts sources et cibles sont mutuellement exclusives. En d'autres termes, il existe une relation de type *Ou Exclusif* entre ces sections. Le choix d'une de ces sections empêche alors la sélection des autres sections du paquet. Graphiquement, le paquet est représenté en pointillé et les différentes stratégies sont mentionnées sous la forme d'une fourche.

La relation de type paquet représente la variabilité dans la réalisation du but cible. Elle utilise l'opérateur « ⊗ » pour formaliser l'exclusivité des sections constituant le paquet. Le

paquet entre deux buts *a* et *b* est notée : $P_{ab} = \otimes (ab_1, ab_2, \ldots, ab_n)$, où les $ab_i \in \{1..n\}$ représentent les sections exclusives et *n* le nombre de ces sections.

Un exemple de paquet est présenté à la Figure 26. Dans cette figure, un paquet, nommé *Par planification*, regroupe deux sections exclusives pour l'acquisition des produits : *Par seuil de réapprovisionnement* ou *Par prévisions stratégiques*. L'utilisation de l'une de ces sections exclut l'utilisation de l'autre.

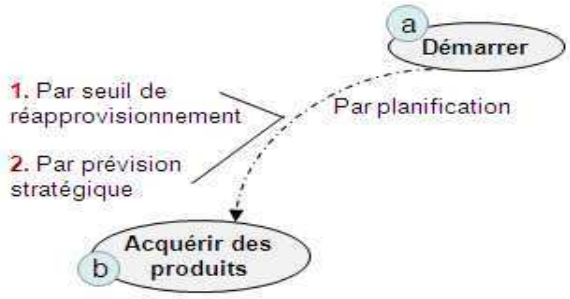

Figure 26. Exemple de paquet

4.4.4.2 *LE MULTI-SEGMENT*

Dans une carte, il est possible d'atteindre un but cible à partir d'un but source selon différentes stratégies. Cette topologie est appelée *multi-segment*. Le multi-segment peut être vu comme une relation logique *Et/Ou* entre les sections ayant le même but source et le même but cible. La relation *Ou* représente l'alternative entre les sections du multi-segment. Celles-ci sont suffisantes seules pour satisfaire le but. La relation *Et*, quant à elle, représente la complémentarité entre les sections du multi-segment. Celles-ci sont acceptables et nécessaires ensemble. L'une d'entre elles ne suffit pas pour satisfaire le but cible.

La relation de type multi-segment représente également la variabilité dans la réalisation du but cible. Elle utilise l'opérateur « v » pour formaliser la relation existante entre les sections du multi-segment. Le multi-segment entre deux buts *a* et *b* est notée : $MS_{ab} = v (ab_1, ab_2, \ldots, ab_n)$, où les $ab_i \in \{1..n\}$ correspondent aux sections complémentaires et *n* le nombre de ces sections.

Dans l'exemple de la Figure 27, les deux sections ab_1, ab_2 et ab_3 forment un multi-segment, car elles représentent deux façons différentes, complémentaires, d'*Acquérir des Produits*.

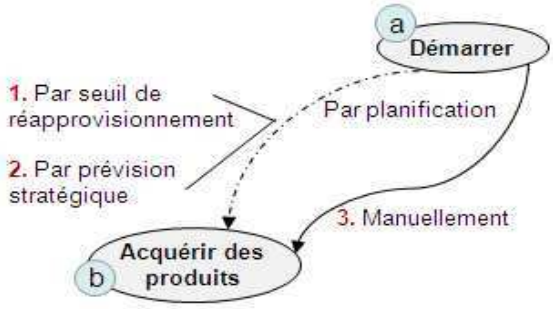

Figure 27. Exemple de multi-segment

4.4.4.3 *LE CHEMIN*

La relation entre les sections d'une carte établit quels buts précèdent ou succèdent tels autres. La relation de précédence indique qu'un but ne peut être réalisé que si un autre but a été réalisé. La relation de précédence conduit à ordonner les sections dans un *chemin*. Dans un chemin, le but cible de la section qui précède est le but source de la section qui lui succède. Un chemin est donc un sous-ensemble de sections de la carte où les sections sont reliées par une relation de précédence. La relation de type chemin utilise l'opérateur cumulatif « ● » pour formaliser la précédence existante entre les sections et les relations entre elles.

Si, à partir d'un but source, il est possible d'atteindre un même but cible en suivant plusieurs chemins de la carte, on parle alors de topologie *multi-chemin*. Le multi-chemin permet de définir les différentes combinaisons de sections possibles pour atteindre un but cible à partir d'un but source. La carte est elle-même multi-chemin, car elle peut contenir plusieurs chemins possibles pour aller de *Démarrer* à *Arrêter*. Elle est alors considérée comme un modèle multiprocessus. En effet, un chemin de la carte représente un processus métier, lequel permet de réaliser un but métier particulier. Le multi-chemin de la carte désigne alors plusieurs processus métier, lesquels permettent de réaliser un même but métier. La relation multi-chemin représente la variabilité cumulative dans la réalisation du but final. La relation de type multi-chemin repose sur l'opérateur d'union « ∪ » pour formaliser la relation existante entre les chemins alternatifs.

Les relations de type chemin et multi-chemin, entre deux buts B_s et B_c, sont notées respectivement par : $C_{Bs,\,Q,\,Bc}$ et $MC_{Bs,\,Q,\,Bc}$, où Q représente l'ensemble des buts intermédiaires empruntés pour la réalisation du but cible B_c à partir du but source B_s.

La Figure 28 présente un exemple de multi-chemin : deux chemins sont mentionnés entre *Démarrer* et *Arrêter*. Le chemin 1 est composé des sections ac_1 et cd_1. Il décrit le besoin de contrôler le stock avant le contrôle du paiement. Le chemin 1 est noté : $C_{a,\,\{c\},\,d} = \bullet(ac_1, cd_1)$. Le chemin 2 est composé des sections ab_3, bc_1 et cd_1. Il décrit le besoin d'abord d'acquérir les

produits, ensuite de contrôler le stock, et enfin de contrôler le paiement. Le chemin 2 est noté : $C_{a,\{b,c\},d} = \bullet(ab_3, bc_1, cd_1)$.

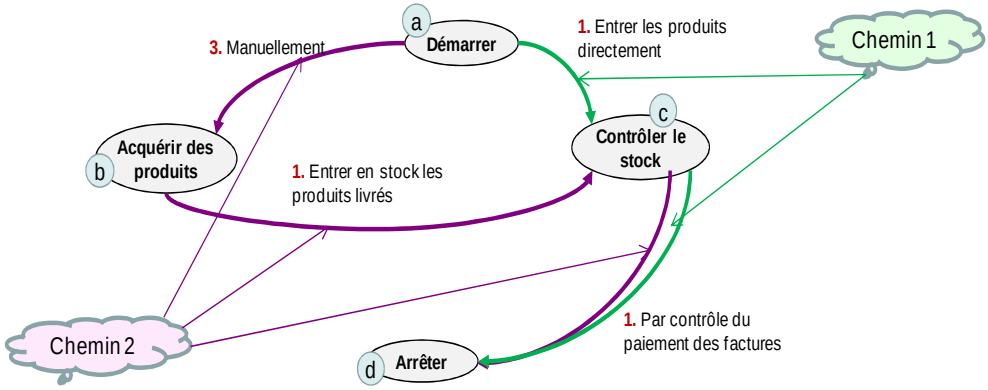

Figure 28. Exemple de chemin et de multi-chemin

4.4.5 LA RELATION D'AFFINEMENT

Le méta-modèle de la Carte présenté à la Figure 24 montre qu'une section peut être décrite par une carte complète, via la relation d'*affinement*. La relation d'affinement permet de décrire une section à un niveau d'abstraction donné (i), par une carte complète à un niveau d'abstraction moins élevé (i+1) (soit à un niveau plus de détail plus fin). Les sections de la nouvelle carte peuvent elles-mêmes être affinées par de nouvelles cartes, jusqu'à obtention des sections opérationnalisables (section ne pouvant plus être affinée). Par exemple, la section bc_1 *<Acquérir des produits, Contrôler le stock, Entrer en stock des produits livrés>* peut être affinée en une carte de nature plus opérationnelle.

Avec la codification absolue, les buts sont nommés *C.a, C.b, etc.* et ses sections $C.ab_1$, $C.ab_2$, $C.bc_1$, *etc.* Si l'une de ces sections (par exemple $C.ab_1$) est affinée par une carte, celle-ci a pour code $C.C_{ab1}$. Par exemple, ses trois buts (*a*, *b* et *c*) et ses sections (ab_1, ab_2, bc_1 et bc_2) sont alors codés de façon absolue comme suit :

- Les buts : $C.C_{ab1}.a$, $C.C_{ab1}.b$ et $C.C_{ab1}.c$
- Les sections : $C.C_{ab1}.ab_1$, $C.C_{ab1}.ab_2$, $C.C_{ab1}.bc_1$, $C.C_{ab1}.bc_2$.

La Figure 29 montre quatre niveaux de granularité, trois niveaux représentés sous forme graphique. La carte *C* de *niveau 0* propose plusieurs sections. Chaque section peut être détaillée dans une carte de *niveau 1*. Par exemple la partie gauche de la figure montre que les sections ab_1 et ab_2 de la carte *C* (notées respectivement $C.ab_1$ et $C.ab_2$) peuvent être affinés dans des cartes de *niveau 1*, notées respectivement $C.C_{ab1}$ et $C.C_{ab2}$. Les sections de ces cartes peuvent également être affinées dans des cartes de *niveau 2*. Par exemple, la section bd_1 de la carte $C.C_{ab1}$ (soit $C.C_{ab1}.bd_1$) peut être affinée dans la carte $C.C_{ab1}.C_{bd1}$, et ainsi de suite.

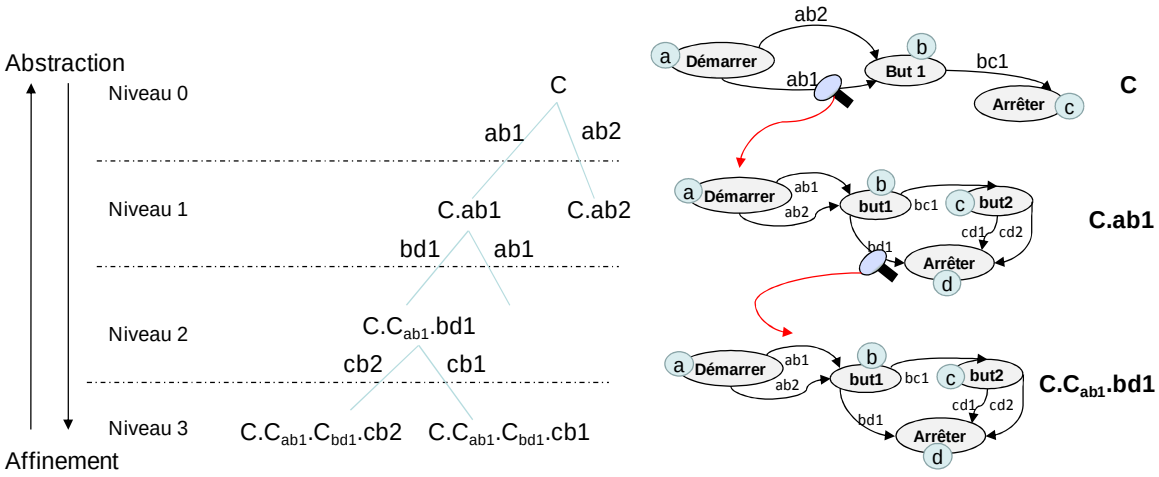

Figure 29. Affinement de section et notation

La relation d'affinement permet une modélisation des besoins à divers niveaux d'abstraction offrant ainsi différentes vues des processus ; allant du niveau le plus stratégique jusqu'au niveau le plus opérationnel.

4.4.6 INVARIANTS ET REGLES DE VALIDITE DE LA CARTE

Dans la perspective de générer des cartes correctes et valides, la construction des cartes doit respecter un certain nombre d'invariants et doit vérifier certaines règles de validité.

4.4.6.1 *INVARIANTS DE LA CARTE*

Un invariant de la carte est une propriété que la carte doit toujours vérifier (Rolland et *al.*,2004) (Banerjee et *al.*,1987). Les trois invariants à vérifier sont :

1. Toute carte possède un seul but qui n'est la cible d'aucune stratégie ; c'est le but D*émarrer* ;
2. Toute carte possède un seul but qui n'est la source d'aucune stratégie ; c'est le but *Arrêter* ; et
3. Tout but de la carte doit se réaliser au minimum une fois, c'est-à-dire qu'il existe un chemin qui le relie au but *Démarrer*.

À partir de ces trois invariants, les cinq corollaires suivants sont déduits :

1. Les cartes sont des graphes connexes ; il n'y a aucun but ou stratégie isolée ;
2. Chaque but dans une carte est la source d'au moins une stratégie excepté le but *Arrêter* ;
3. Chaque but dans une carte est la cible d'au moins une tactique excepté le but *Démarrer* ;
4. Il y a toujours un chemin de *Démarrer* à *Arrêter* ; et

5. Chaque section d'une carte appartient à au moins un chemin entre *Démarrer* et *Arrêter*.

4.4.6.2 *REGLES DE VALIDITE DE LA CARTE*

Afin qu'une carte soit valide, elle doit vérifier les règles suivantes (Rolland&Prakash,2001) :

1. Les buts et stratégies d'une carte donnée sont à un même niveau d'abstraction ;
2. Aucun but et stratégie d'une carte ne doit pouvoir être considéré comme un sous-ensemble d'une autre carte ;
3. Aucun but ne doit apparaître dans une carte comme une manière d'en réaliser une autre ;
4. Les buts ayant pour résultat la même partie de produit doivent être agrégés ;
5. Les sections représentant des manières mutuellement exclusives de produire un même résultat doivent être regroupées en paquet ;
6. Les buts considérés comme faisant partie d'une transaction (tout ou rien) doivent être abstraits sous la forme d'un unique but ; et
7. Les buts qui se complémentent mutuellement et vont ensemble doivent être agrégées au sein d'un unique but.

4.5 *REGLES DE GENERATION DES SERVICES INTENTIONNELS*

Le modèle de la Carte (Rolland&Prakash,2000) est utilisé dans l'approche iSOA (Kaabi et *al.*,2004) (Kaabi,2007) pour modéliser le métier des organisations et les besoins des utilisateurs. L'une des caractéristiques du modèle de la Carte est sa double facette : elle permet de combiner dans une unique représentation la perspective métier et la perspective système (Salinesi&Rolland,2003). La perspective métier correspond aux buts et aux stratégies pour les atteindre et la perspective système définit les services du système qui sont capables de supporter la réalisation de ces buts.

Cette section permet de présenter l'ensemble des règles méthodologiques permettant de générer les différents types de services du modèle *MiS* (*i.e.* service atomique et service agrégat) à partir d'une carte intentionnelle (Kaabi,2007).

4.5.1 GENERATION DES SERVICES ATOMIQUES

Afin d'établir un couplage direct entre les buts métier et les fonctionnalités du système, un service atomique du système correspond à une section opérationnalisable de la carte (Kaabi et

al.,2004) (Kaabi,2007). Dans l'exemple de la carte *Effectuer l'approvisionnement en produits* de la Figure 25, plusieurs services atomiques sont identifiés et récapitulés au Tableau 5.

Tableau 5. Les services atomiques de la carte *Effectuer l'approvisionnement en produits*

Section de la carte	Service atomique
ab_1	$S_{\text{Acquérir des produits par seuil de réapprovisionnement}}$
ab_2	$S_{\text{Acquérir des produits par prévisions stratégiques}}$
ab_3	$S_{\text{Acquérir des produits manuellement}}$
ac_1	$S_{\text{Entrer des produits en stock directement}}$
cc_1	$S_{\text{Transférer des produits}}$
cc_2	$S_{\text{Valuer le stock}}$
cc_3	$S_{\text{Inspecter la qualité du stock}}$
cc_4	$S_{\text{Etablir un inventaire en continu}}$
cc_5	$S_{\text{Etablir un inventaire par échantillonnage}}$
cc_6	$S_{\text{Etablir un inventaire périodiquement}}$
cd_1	$S_{\text{Contrôler le paiement des factures}}$

4.5.2 GENERATION DES SERVICES AGREGATS

Généralement, la carte des besoins représente différentes combinaisons possibles, lesquelles sont exprimées par les relations de paquet, de multi-segment, de chemin et de multi-chemin. Chaque combinaison à partir de *Démarrer* jusqu'à *Arrêter* correspond à un chemin.

Néanmoins, il est difficile d'identifier toutes les combinaisons possibles d'une carte. Ainsi, l'algorithme de MacNaughton et Yamada (MacNaughton&Yamada,1960) est utilisé pour générer automatiquement tous les chemins d'une carte (Kaabi,2007). L'algorithme se base sur deux formules principales.

La première formule est : $Y_{s,Q,t} = \bullet(X^*_{s,\ Q\setminus\{s\},\ s},\ X_{s,\ Q\setminus\{s,t\},\ t}\ ,\ X^*_{t,\ Q\setminus\{s,t\},\ t}\ )$. $Y_{s,Q,t}$ indique l'ensemble des chemins possibles entre le but *Démarrer* et *Arrêter* de la carte, notés respectivement *s* et *t*.

La deuxième formule est : $X_{s,P,t} = \cup((X_{s,\ P\setminus\{q\},\ t}\ ),\ \bullet(X_{s,\ P\setminus\{q\},\ q},\ X^*_{q,\ P\setminus\{q\},\ q},\ X_{q,\ P\setminus\{q\},\ t}\ )$. Elle désigne les chemins possibles du but *s* vers le but *t*, en passant particulièrement par un but intermédiaire quelconque *q*. Chaque $X_{s,P,t}$ est développé à son tour en utilisant la formule précédente, en prenant, au hasard, un but intermédiaire de l'ensemble *P* (*i.e.* l'ensemble des buts excluant *s* et *t*). Le processus s'arrête lorsque l'ensemble *P* est vide.

La première formule est appliquée sur la carte *Effectuer l'approvisionnement en produits* de la Figure 25, nous obtenons : $Y_{a,\{a,b,c,d\},d} = \bullet(X^*_{a,\{b,c,d\},a},\ X_{a,\{b,c\},d}\ ,\ X^*_{d,\{b,c\},d})$. La deuxième formule est appliquée pour chaque $X_{s,P,t}$. La formule suivante est obtenue : $Y_{a,\{a,b,c,d\},d} = \bullet(\cup(X_{ac},\ \bullet(X_{ab},\ X_{bc})),\ X^*_{cc}\ ,\ X_{cd}\ ))$.

Dans une carte, une section non-opérationnalisable décrite à un niveau d'abstraction donné (*i*), est affinée par une carte à un niveau d'abstraction moins élevé (*i+1*) (*cf.* Section 4.4.5). Le résultat obtenu au niveau (*i+1*) est alors transféré au niveau (*i*), suivant un raisonnement Botton-up. Par exemple, supposons que la section <*Acquérir des produits*, *Contrôler le stock*, *Entrer en stock des produits livrés* > (ayant le code bc_1), de la carte *Effectuer l'approvisionnement en produits* (*cf.* Figure 25) est une section non-opérationnalisable. Celle-ci est affinée par la carte de la Figure 30. L'algorithme de MacNaughton et Yamada est appliqué à cette nouvelle carte, et le résultat obtenu est défini par la formule suivante : Y_{Cbc1} = ●(C_{bc1}.X_{ab}, ∪ (C_{bc1}.X_{bd}, ● (C_{bc1}.X_{bc}, C_{bc1}.X_{cd}))). Il s'agit alors de remplacer la section non-opérationnalisable C_{bc1} par la formule Y_{Cbc1}. La formule de la carte *Effectuer l'approvisionnement en produits* (*cf.* Figure 25) est donc la suivante : $Y_{a,\{a,b,c,d\},d}$ = ●(∪(X_{ac},●(X_{ab},●(C_{bc1}.X_{ab},∪(C_{bc1}.X_{bd},●(C_{bc1}.X_{bc}, C_{bc1}.X_{cd}))))),X^*_{cc}, X_{cd})).

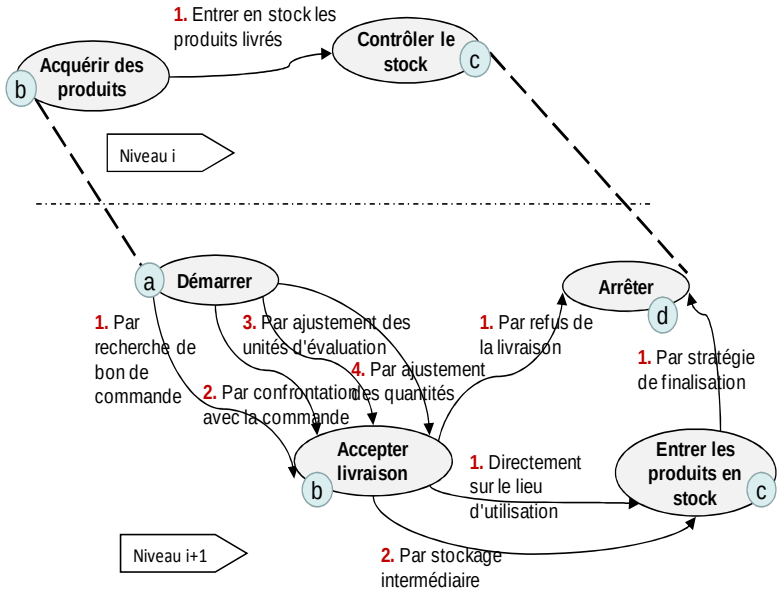

Figure 30. Affinement de la section <*Acquérir des produits*, *Contrôler le stock*, *Entrer en stock des produits livrés* >

Les formules obtenus de type X_{st} sont affectées à une section ou aux différents types de relation existant entre les sections de la carte, à savoir : le paquet (*P*), le multi-segment (*MS*), le chemin (*C*) et le multi-chemin (*MC*). Par exemple, en vérifiant la carte *Effectuer l'approvisionnement en produits* de la Figure 25, nous remarquons que la formule X_{ab} est affectée au multi-segment $MS_{ab} = v$ (P_{ab}, ab_3).

Un service agrégat correspond à chaque type de relation entre les sections de la carte (*P*, *MS*, *C*, *MC*). Le service composite correspond à la relation de type *Et* entre les sections de la carte, à savoir le chemin, tandis que le service à variation correspond à la relation de type

Ou/Ou exclusif entre les sections de la carte, c'est-à-dire le paquet, le multi-segment et le multi-chemin. Un service à choix alternatif correspond à un paquet de la carte, par exemple $\mathbf{S}_{\text{Acquérir des produits par planification}} = \otimes(\mathbf{S}_{\text{Commander des produits par seuil de réapprovisionnement}}, \mathbf{S}_{\text{Commander des produits par prévisions stratégiques}})$. Un service à choix multiple correspond à un multi-segment de la carte, par exemple $\mathbf{S}_{\text{Acquérir des produits}} = \nu(\mathbf{S}_{\text{Commander des produits manuellement}}, \mathbf{S}_{\text{Acquérir des produits par planification}})$. Un service composite séquentiel correspond à un chemin de la carte, par exemple $\mathbf{S}_{\text{Entrer les produits en stock normalement}} = \bullet(\mathbf{S}_{\text{Acquérir des produits}}, \mathbf{S}_{\text{Entrer en stock les produits livrés}})$. Un service à variation de chemin correspond à un multi-chemin de la carte, par exemple $\mathbf{S}_{\text{Recevoir le stock}} = \cup(\mathbf{S}_{\text{Entrer les produits en stock directement}}, \mathbf{S}_{\text{Entrer les produits en stock normalement}})$.

4.6 *CONCLUSION*

L'architecture orientée service intentionnel (*iSOA-intentional Service Oriented Architecture*) représente la transposition de l'architecture SOA au niveau intentionnel. Elle permet de décrire les services de façon intentionnelle, en mettant en avant le but que le service permet de réaliser. Le but du service joue un rôle important lorsque l'agent métier souhaite rechercher le service correspondant à son intention.

Dans l'architecture iSOA, le modèle *MiS* permet de représenter, à un niveau intentionnel, les services du système. Le service intentionnel peut être un service atomique ou un service agrégat. Le service atomique est opérationnalisé par un ensemble d'actions, lequel est modélisé par un service logiciel ou une composition de services logiciels. Par conséquent, le service atomique est exécutable. Le service agrégat, quant à lui, reflète la variabilité dans la réalisation du but du service, à travers le service à variation de chemin, le service à choix multiple et le service à choix alternatif.

Nous adoptons l'architecture iSOA dans cette thèse car elle permet de diminuer la discordance conceptuelle entre les exigences fonctionnelles formulées par les agents métier et la définition des services logiciels. En effet, le service intentionnel est un service dédié à l'utilisateur final, sans autant omettre la perspective logicielle de ce service.

Néanmoins, le problème de discordance conceptuelle entre l'expression des exigences non-fonctionnelles et la description de la qualité des services logiciels est toujours persistant. Pour résoudre ce problème, nous proposons de définir une extension du modèle *MiS* à travers la notion de qualité de service intentionnel (*QoiS-Quality of intentional Service*) (*cf.* Chapitre 6), ainsi que de la méthodologie de génération de cette qualité (*cf.* Chapitre 7).

CHAPITRE 5 : REFERENTIEL QUALITE

5.1 *INTRODUCTION*

Ce chapitre présente le méta-modèle du référentiel qualité. Comme présenté dans les chapitres précédents, l'originalité de la méthode de sélection intentionnelle, que nous proposons dans cette thèse, repose sur une expression unifiée des exigences non-fonctionnelles et de la qualité des services logiciels.

Dans ce chapitre, nous définissons formellement la modélisation de la qualité de service, à travers un méta-modèle nommé *référentiel qualité*, lequel est composé de deux parties principales :

- La partie méta-modèle qui explicite les buts qualité et leurs décompositions.
- La partie métrique qui complète cette première partie en at tribuant des mesures quantitatives à la qualité de service.

Le référentiel qualité permet, pour un domaine d'activité particulier, de définir un catalogue de qualité contenant les qualités et leurs valeurs de satisfaction. Il est utilisé, conjointement, par les fournisseurs et les agents (ou clients) métier : les premiers pour publier la qualité de leur service et les derniers pour formuler leurs souhaits en terme de qualité.

Il est largement admis par les deux communautés SOC (Maximilien&Singh,2004b)(Dobson et *al.*,2005)(Wang et *al.*,2006) et RE (Penserini et *al.*,2006) (Ma et *al.*,2009) que la réussite de la sélection de service nécessite la définition d'un consensus qualité entre les clients et les fournisseurs. Dans l'architecture iSOA, nous proposons un référentiel qualité permettant, pour un domaine d'activité particulier, de définir un catalogue de qualité contenant les qualités et leurs valeurs de satisfaction. Il est utilisé, conjointement, par les fournisseurs et les agents (ou clients) métier : les premiers pour publier la qualité de leur service et les derniers pour formuler leurs souhaits en terme de qualité. Le référentiel qualité permet alors de fournir un consensus qualité entre les fournisseurs et les clients. Il est utilisé par :

Les fournisseurs d'un même domaine : le référentiel qualité est le catalogue permettant aux différents fournisseurs de partager les mêmes connaissances concernant la qualité de service. En effet, les fournisseurs, lors de la publication de la qualité de leurs services, peuvent manquer d'objectivité, en surestimant leurs qualités. Par exemple, un fournisseur peut

publier un service de réservation de billet d'avion en estimant que son service propose des billets pas chers avec des compagnies aérienne très fiables et en proposant un paiement très sécurisé. Un autre fournisseur peut publier un même service avec les mêmes qualités. Cependant, les deux fournisseurs n'utilisent pas forcément le même vocabulaire et ils ne donnent pas nécessairement la même signification aux valeurs de satisfaction, car ce qui est « pas cher », « très fiable » et « très sécurisé » pour l'un ne l'est pas forcément pour l'autre. Par conséquent, il est nécessaire d'uniformiser le vocabulaire qualité (les qualités et leurs évaluations) entre les différents fournisseurs d'un même domaine. De plus, les termes utilisés par les fournisseurs métier, tels que « rapide », « très rapide » et « sécurisé », doivent être mis en correspondance avec les valeurs de qualité délivrées par les services logiciels. Par exemple, « très rapide » correspond à la valeur de référence « temps de réponse < *3* minutes ». Néanmoins, ces valeurs peuvent être différentes d'un fournisseur à un autre et elles dépendent du domaine en question. Il est donc nécessaire d'avoir un consensus, entre les fournisseurs d'un domaine, sur les valeurs de référence correspondant à « très rapide », « rapide » et « sécurisé », par exemple.

Ainsi, le référentiel qualité permet de diminuer l'écart existant entre la qualité délivrée par les services logiciels et les exigences de qualité des clients. Il permet également de donner une sémantique moins subjective aux valeurs de satisfaction. Le référentiel qualité permet d'améliorer la crédibilité des fournisseurs, car les fournisseurs d'un domaine utilisent le même référentiel pour décrire la qualité de leurs services. Aussi, il est un bon moyen de réutilisation des qualités, car il évite aux fournisseurs de construire leurs propres référentiels.

Les fournisseurs et les clients d'un même domaine : le référentiel qualité est le catalogue permettant aux fournisseurs et aux clients de partager les mêmes concepts et les mêmes valeurs de satisfaction concernant la qualité de service. En effet, lorsque les clients émettent leurs besoins en termes de qualité (*i.e.* leurs exigences non-fonctionnelles), ils doivent s'aligner aux descriptions des qualités de services, publiées par les fournisseurs. Par exemple, le client X peut utiliser des seuils de satisfaction quantitatifs (99%, 13 secondes, 100 euros) ou linguistique (bon, très bon, satisfait, *etc*.) qui peuvent ne pas correspondre aux qualités publiées par les fournisseurs. De plus, les valeurs de satisfaction définies par les fournisseurs peuvent être perçues différemment par les divers clients. Par conséquent, il est nécessaire de partager un langage commun concernant la qualité (les qualités et leurs évaluations) entre les clients et les fournisseurs du même domaine.

Ainsi, le référentiel qualité permet d'établir la mise en correspondance au niveau intentionnel entre les exigences non-fonctionnelles des agents métier et la qualité de service,

établie par le fournisseur métier. Il permet également de diminuer le risque d'incompatibilité lors de la sélection des services.

Ce chapitre est composé de plusieurs sections principales. La section 2 introduit la vue globale du méta-modèle du référentiel qualité. La section 3 détaille les différents concepts de ce modèle. La section 4 présente une instanciation du méta-modèle du référentiel qualité, tandis que la section 5 montre la manière de construire ce référentiel. Enfin, la section 6 conclut ce chapitre.

5.2 *VUE GLOBALE DU REFERENTIEL QUALITE*

Le méta-modèle du référentiel qualité est présenté à la Figure 31 en utilisant la notation UML[4]. Le *but qualité* est l'élément clé du méta-modèle. Nous nous inspirons du cadre d'application NFR (*NFR framework*) (Mylopoulos et *al.*,1992)(Chung et *al.*,2000) pour construire notre référentiel qualité. En effet, une partie du méta-modèle (but qualité, décomposition, corrélation et seuil de satisfaction) reprend les concepts du cadre NFR, lequel permet de traiter les exigences non-fonctionnelles (*NFR-Non Functional Requirements*) comme des buts qualité à satisfaire partiellement. La deuxième partie du méta-modèle consiste à étendre ce cadre avec des valeurs de référence concernant la satisfaction partielle des buts qualité (métrique, classe, *etc.*).

Le référentiel qualité permet, d'une part, de promouvoir la réutilisation de la qualité dans l'architecture iSOA et d'autre part de fournir un consensus qualité entre les fournisseurs et les agents métier. Le référentiel qualité est défini par domaine d'activité. Il est possible de retrouver les mêmes qualités dans deux domaines différents, mais avec des opérationnalisations, des moyens de mesure et des valeurs de satisfaction différents.

La Figure 31 illustre les concepts du référentiel qualité. Ces derniers mettent en évidence trois principaux aspects, à savoir :

- les aspects coloriés en gris foncé à la Figure 31, lesquels concernent la définition du but qualité, son affinement, *etc.* ;
- les aspects coloriés en gris clair à la Figure 31, lesquels définissent l'évaluation qualitative de la satisfaction partielle (*i.e.* le *satisficing*) du but qualité. Celle-ci est représentée par le but, les liens de contribution et les seuils de satisfaction ; et

[4] Unified Modelling Language: www.uml.org/

- les aspects coloriés en blanc à la Figure 31, lesquels correspondent à l'évaluation quantitative de la satisfaction du but qualité. Celle-ci est définie par les métriques, les classes, *etc*.

Nous utilisons dans le référentiel qualité les *buts* (But à la Figure 31) pour représenter les *hard goals* qui contribuent à satisfaire partiellement les buts qualité. Comme défini dans l'état de l'art, le *hard goal* est le but dont la satisfaction est objective, car ses critères de réalisation sont clairement définis. Il est dit réalisable contrairement au but qualité (But qualité à la Figure 31) qui est dit « *satisficeable* » (*cf.* Section 2.2.5.1.1). Le référentiel qualité contiendra, par domaine, l'ensemble des buts qualité potentiels, ainsi que les buts contribuant à les satisfaire partiellement. Par exemple, dans le domaine du diagnostic *in vitro* de l'hémostase, le but « Mesurer plasma par photométrie » contribue à satisfaire négativement (« - ») la performance (Lien contribution et Seuil de satisfaction à la Figure 31), tandis que le but « Mesurer plasma par chronométrie » contribue à satisfaire positivement (« + ») cette dernière. Ces seuils de satisfactions sont définis à partir des plages de valeurs, dites classe (Classe à la Figure 31) dans le référentiel qualité. En effet, dans le référentiel qualité, les buts qualité peuvent être quantifiés par des métriques (Métrique à la Figure 31), lesquelles permettent d'évaluer la qualité des conteneurs logiciels opérationnalisant les buts. Les valeurs obtenues sont alors regroupées en classe et chaque classe est associée à un seuil de satisfaction (association équivaut entre Classe et Seuil de satisfaction à la Figure 31). La classe permet alors de faire le lien entre les seuils qualitatifs de satisfaction (« ++ », « + », « - », « -- ») et les valeurs obtenues suite à l'exécution des conteneurs logiciels.

Dans iSOA, le référentiel qualité est consulté d'une part, par les fournisseurs métier pour décrire la QoiS de leurs services intentionnels (cf. Chapitre 6 et Chapitre 7), et d'autre part par les agents métier pour décrire leur contexte qualité (cf. Chapitre 8). Le service permet de réaliser un but, alors que sa QoiS correspond à l'ensemble des buts qualité que ce service permet de satisfaire partiellement. Autrement dit, le fournisseur métier publie des services réalisant un but, et en consultant le référentiel qualité de son domaine, le fournisseur compare le but de son service aux buts du référentiel, afin de déduire les buts qualité que ses services peuvent satisfaire partiellement.

Une vue détaillée de ces différents aspects est présentée dans la suite de ce chapitre.

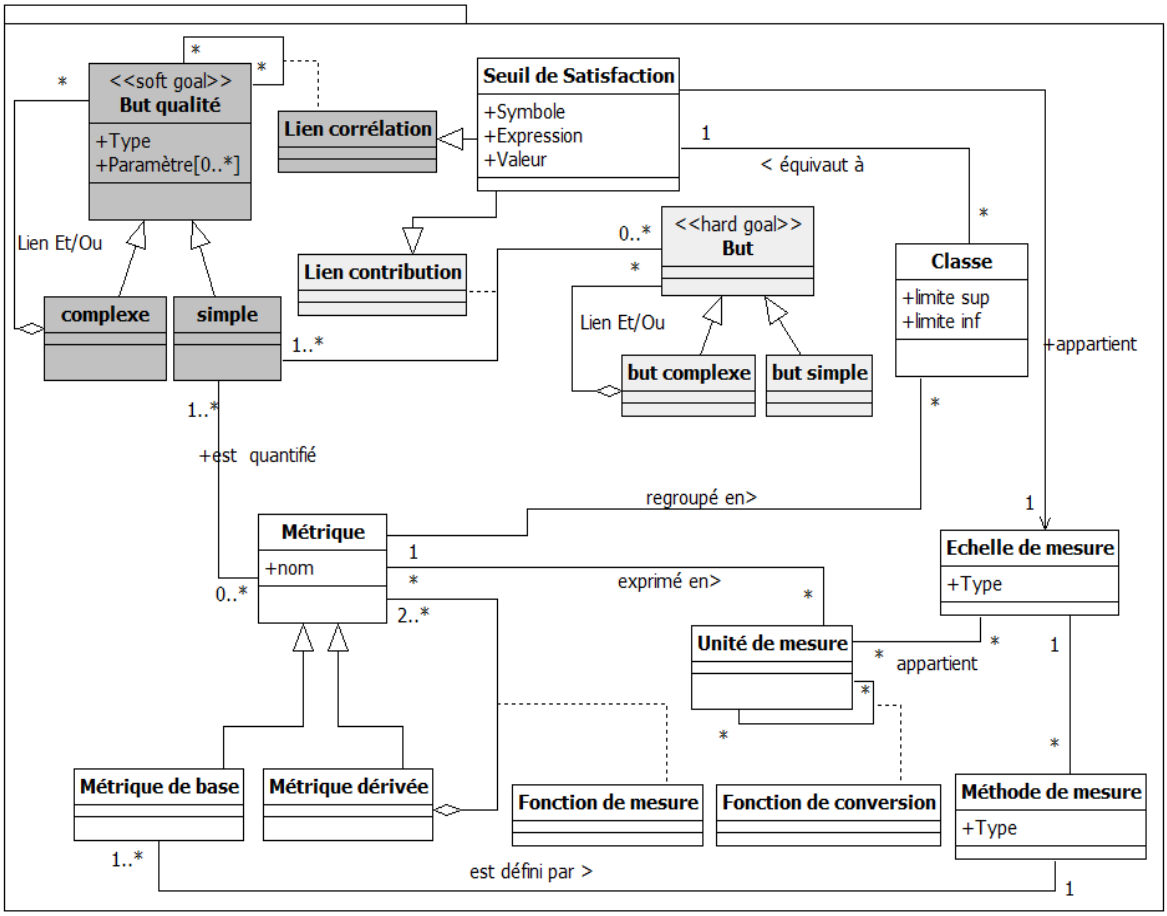

Figure 31. Le méta-modèle du référentiel qualité

5.3 *VUE DETAILLEE DU REFERENTIEL QUALITE*

Afin de bien comprendre le référentiel qualité, il est important de préciser la signification de chacun de ses concepts.

5.3.1 DEFINITIONS DU BUT QUALITE

Dans cette sous-section, nous présentons la définition des aspects coloriés en gris dans la Figure 31, en commençant par celle du but qualité.

5.3.1.1 *LE BUT QUALITE*

Comme présenté dans l'état de l'art (*cf.* Chapitre 2), le but qualité représente les exigences non-fonctionnelles, telles que la performance, la précision et la sécurité. Les buts qualité sont interdépendants et peuvent être affinés en des sous-buts qualité dont l'abstraction est plus faible. Les éléments du méta-modèle du référentiel qualité permettant définir le but qualité, sa typologie (simple ou complexe) et le lien de corrélation entre les buts qualité sont présentés à la Figure 32.

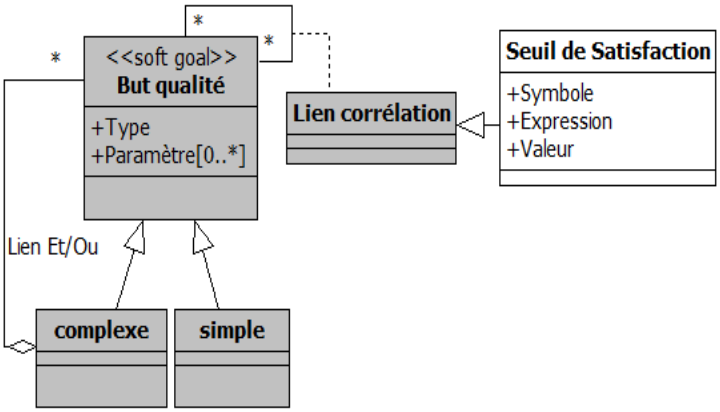

Figure 32. Le but qualité

Dans le référentiel qualité, le but qualité reflète d'une part les exigences non-fonctionnelles que le service doit satisfaire, et d'autre part les besoins de qualité des clients. Le but qualité est caractérisé par une satisfaction subjective et relative, laquelle est modélisée via la notion de *satisficing*. Celle-ci permet d'exprimer qu'un but qualité peut être satisfait dans des limites acceptables, plutôt que de façon absolue. Il convient alors d'un accord entre les différents partenaires du service (clients et fournisseurs) pour définir des seuils de satisfaction, permettant de décider à partir de quel moment le but qualité est considéré partiellement satisfait. En d'autres termes, la satisfaction de ces buts tient à l'établissement de ces seuils et l'acceptation par le client de tout aboutissement au-delà de ces seuils.

Nous reprenons la notation et la représentation graphique du but qualité proposées par le cadre NFR (*cf.* Chapitre 2). Tel que présenté à la Figure 32, le but qualité est décrit par un *type* et un ensemble de *paramètres*, notés *Type[Paramètre]*. Le type correspond à la catégorie de NFR, telle que la sécurité, la performance, la précision, *etc*. Le paramètre est optionnel et il correspond au sujet concerné par le type. Autrement dit, le paramètre correspond à la cible qui a besoin de satisfaire une qualité particulière. Par exemple, dans un système de réalisation d'analyse médicale, on a besoin que le résultat de l'analyse obtenu soit précis, d'où le but qualité Précision[RésulatAnalyse]. Aussi, on a besoin que l'analyse soit faite rapidement, d'où Performance[Analyse].

Un autre exemple de but qualité concerne la confidentialité des données de paiement d'une réservation. Ce but qualité est alors noté : Confidentialité [DonnéesPaiementRéservation]. La représentation graphique du but qualité est définie par un cercle étiqueté avec le nom du but qualité. La Figure 33 montre la représentation graphique du but qualité Confidentialité[DonnéesPaiementRéservation].

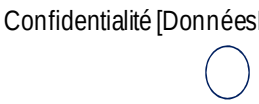

Figure 33. Exemple de représentation graphique d'un but qualité.

5.3.1.2 *L'AFFINEMENT DU BUT QUALITE*

Comme présenté dans l'état de l'art (*cf.* Chapitre 2), l'affinement du but qualité peut se faire par des liens d'affinement *Et/Ou*. Dans le référentiel qualité, nous définissons deux types de but qualité (*cf.* Figure 32) : un but qualité *simple* et un but qualité *complexe*. Un but qualité est dit complexe lorsqu'il peut être affiné (*i.e.* décomposé), par des liens d'affinement *Et/Ou*, en d'autres sous-buts qualité, dont l'abstraction est plus faible. Dans ce cas, le type et le paramètre du but qualité peuvent être affinés. Le but qualité complexe est affiné jusqu'à rencontrer des buts qualité simples. Un but qualité est dit simple lorsque sa satisfaction partielle peut être effectuée par des buts dont la satisfaction est précise (*i.e. hard goal*). Les buts qualité, tels que présentés à la Figure 34, peuvent être représentés par le graphe de buts qualité (*cf.* Chapitre 2), lequel permet de relier les buts qualité par des liens d'affinement *Et/Ou*.

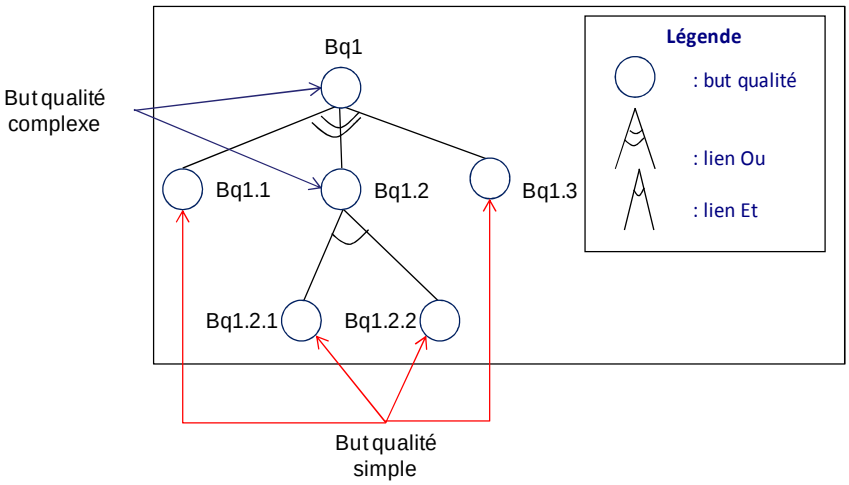

Figure 34. Représentation de l'affinement du but qualité

La Figure 35 illustre un exemple d'affinement des buts qualité complexes Sécurité[Réservation] et Rapidité[Réservation]. Le premier affinement est de type *Ou* (*cf.* Figure 35-(a)), il définit que la sécurité des données d'une réservation (*i.e.* Sécurité[Réservation]) consiste à garantir la confidentialité des données de paiement (*i.e.* Confidentialité[DonnéesPaiementRéservation]) ou à garantir l'intégrité des données de réservation (*i.e.* Intégrité[DonnéesRéservation]). Le second affinement est de type *Et* (*cf.* Figure 35-(b)), il indique que garantir la rapidité d'une réservation (*i.e.* Rapidité[Réservation]) consiste à la fois à garantir la rapidité dans la recherche d'une réservation (*i.e.*

Rapidité[Réservation]) et à garantir la rapidité du paiement de la réservation (*i.e.* Rapidité[Paiement]). Ceci correspond au lien *Et*/*Ou* entre les éléments But qualité et But qualité complexe, représentés à la Figure 32.

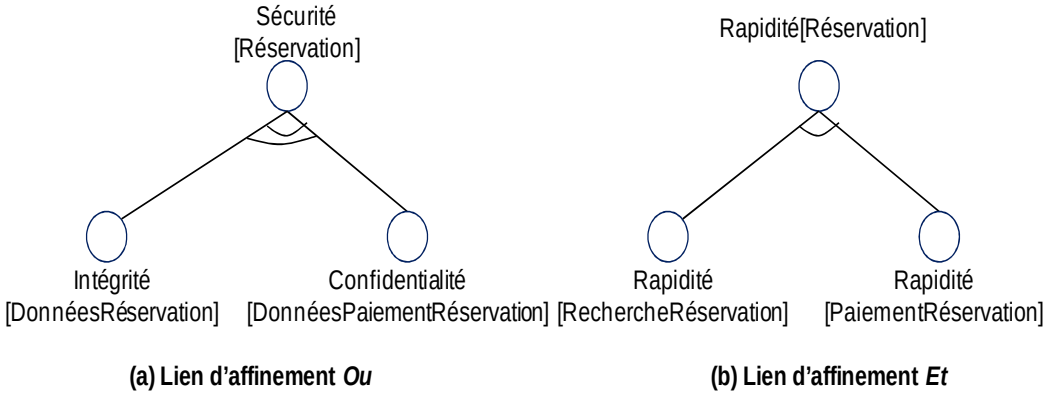

Figure 35. Exemple de décomposition du but qualité

5.3.1.3 *LA CORRELATION ENTRE BUTS QUALITE*

Les buts qualité peuvent être conflictuels ou synergiques (*cf.* Chapitre 2). Un but qualité peut donc influencer négativement (ou positivement) un (ou plusieurs) autre(s) but(s) qualité. Cette influence est représentée par l'élément *lien corrélation* à la Figure 32. Le lien de corrélation tient ses valeurs des seuils de satisfaction. L'influence positive implique que les buts qualité ne sont pas conflictuels, alors que l'influence négative signifie que les buts qualité ne peuvent pas être satisfaits simultanément.

Dans le référentiel qualité, il est important de mettre l'accent sur les qualités conflictuelles. Celles-ci préviennent que les services, qui potentiellement vont satisfaire partiellement une qualité, peuvent ne pas satisfaire partiellement d'autres. Par exemple, la Sécurité et la Performance sont des buts qualité qui se corrèlent négativement : si un service satisfait un niveau important de sécurité (en implémentant des algorithmes de cryptage, par exemple), alors la performance de ce service tendra à réduire (le temps de réponse du service augmente). Ce service contribue alors positivement à la satisfaction de la sécurité et négativement à celle de la performance. En conséquence, ce service ne peut pas satisfaire la Sécurité et la Performance en même temps. La Figure 36 illustre cet exemple de corrélation négative entre les buts qualité Sécurité et Performance.

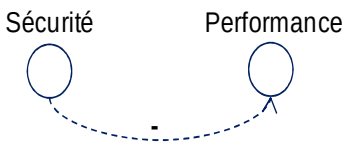

Figure 36. Exemple de corrélation de but qualité

Il est donc important que le référentiel qualité informe des qualités conflictuelles, particulièrement pour les agents métier. En effet, lors de la formulation de leurs préférences en termes de qualité, les agents métier doivent être informés qu'ils ne peuvent satisfaire simultanément certaines qualités (sécurité et performance, par exemple), afin que ces agents puissent mieux préciser leurs préférences en termes de priorisation (*cf.* Chapitre 8).

5.3.2 DOMAINE

L'usage, l'évaluation et l'opérationnalisation d'un but qualité dépendent fortement d'un domaine d'activité auquel il se réfère. De façon générale, les buts qualité, tels que la sécurité, la précision et la performance sont génériques et peuvent être considérés dans n'importe quel domaine. Cependant, l'usage, l'évaluation et l'opérationnalisation d'un but qualité changent d'un domaine à un autre.

En effet, l'opérationnalisation, correspondant aux buts mis en place permettant de satisfaire partiellement les buts qualité, change d'un domaine à un autre. Par exemple, on considère le but qualité Précision[Résultat]. Celui-ci peut être considéré dans plusieurs domaines, tels que le commerce électronique ou le diagnostic *in vitro* de l'hémostase. Cependant, les buts mis en place pour satisfaire partiellement cette qualité dans ces domaines sont différents. Par exemple, les buts « relancer analyse » et « valider résultat » contribuent à la satisfaction partielle de la Précision[Résultat] dans le diagnostic *in vitro* de l'hémostase. Dans le domaine du commerce électronique, le but « confirmer les données » permet lui aussi de satisfaire partiellement la précision. Le référentiel qualité répertorie alors, selon le domaine en question, les buts (*e.g.* « relancer analyse », « valider résultat », « confirmer les données » ...) qui permettent de contribuer positivement ou négativement à la satisfaction des buts qualité potentiels de ce domaine.

De plus, les métriques permettant d'évaluer la satisfaction change selon le domaine considéré. Par exemple, lorsqu'on considère le but qualité Performance, celui-ci peut être aussi requis dans plusieurs domaines, tels que le commerce électronique et dans le domaine du diagnostic *in vitro* de l'hémostase. Par contre, les métriques mises en place pour mesurer la satisfaction de la Performance divergent en fonction du domaine. En effet, pour mesurer la performance dans le domaine du commerce électronique on peut utiliser le « nombre de transaction/h », alors que dans celui du diagnostic *in vitro* de l'hémostase, on peut, par exemple, utiliser la « cadence/h ».

Ainsi, l'affinement du but qualité se fait en fonction du domaine d'activité considéré (correspond au Package à la Figure 31). Au lieu que chaque fournisseur métier construise son

propre catalogue qualité, nous proposons que les fournisseurs d'un même domaine se concertent pour crée un catalogue qualité commun.

Nous proposons de regrouper les fournisseurs métier suivant les domaines d'activité considérés dans la Nomenclature d'Activités Française révision 2 (NAF rév. 2,2008). La NAF (NAF rév. 2,2008) est une nomenclature statistique nationale de toutes les activités économiques (ou industrielles). La NAF se base sur la nomenclature standard des activités industrielles établie par la division statistique des nations unies (*ISIC* rev.4-*International Standard Industrial Classification Revision 4.0*) (ISIC Rev. 4.0,2008). La classification des activités économiques est effectuée selon les activités qui font l'économie des pays. La NAF est utilisée non seulement dans le monde industriel mais également pour faire des statistiques. Le Tableau 6 montre la liste des activités du Niveau 1 de la NAF. Il existe 21 catégories d'activités. Celles-ci sont codées par une lettre allant de *A* jusqu'à *U*, lesquelles sont subdivisée en plusieurs sous-catégories.

Ainsi, les fournisseurs métier d'une catégorie ou d'une sous-catégorie peuvent se rassembler pour construire leur référentiel qualité.

Tableau 6. Nomenclature d'Activités Française

N°	Code	Libellé
1	A	Agriculture, sylviculture et pêche
2	B	Industries extractives
3	C	Industrie manufacturière
4	D	Production et distribution d'électricité, de gaz, de vapeur et d'air conditionné
5	E	Production et distribution d'eau ; assainissement, gestion des déchets et dépollution
6	F	Construction
7	G	Commerce ; réparation d'automobiles et de motocycles
8	H	Transports et entreposage
9	I	Hébergement et restauration
10	J	Information et communication
11	K	Activités financières et d'assurance
12	L	Activités immobilières
13	M	Activités spécialisées, scientifiques et techniques
14	N	Activités de services administratifs et de soutien

15	O	Administration publique
16	P	Enseignement
17	Q	Santé humaine et action sociale
18	R	Arts, spectacles et activités récréatives
19	S	Autres activités de services
20	T	Activités des ménages en tant qu'employeurs ; activités indifférenciées des ménages en tant que producteurs de biens et services pour usage propre
21	U	Activités extraterritoriales

5.3.3 EVALUATION QUALITATIVE DE LA SATISFACTION DU BUT QUALITE

Nous définissons l'évaluation du but qualité comme une mesure (ou une appréciation) quantitative ou qualitative de la satisfaction de ce but, en fonction d'une décision à prendre (par exemple, la sélection de service). La mesure est définie par Fenton et Pfleeger (Fenton&Pfleeger,2004) comme étant le processus à travers lequel, dans le monde réel, des numéros ou des symboles sont affectés aux attributs des entités, de manière à les décrire selon des règles bien définies.

La caractéristique principale des buts qualité est leur satisfaction subjective qui ne peut être définie de façon précise, car elle dépend fortement du contexte des agents métier. Le but qualité se base sur le concept de *satisficing* (*i.e.* accepter la satisfaction) qui consiste à définir un ensemble de seuils de satisfaction et à ce que le client accepte tout aboutissement aux delà de ces seuils.

Dans le référentiel qualité, nous proposons d'évaluer qualitativement la satisfaction partielle du but qualité. Celle-ci se fait via les seuils de satisfaction, lesquels sont compréhensibles par les agents métier. Les éléments de l'évaluation qualitative sont coloriés en gris clair à la Figure 31.

L'évaluation qualitative de la satisfaction partielle des buts qualité consiste à définir un ensemble de seuils de satisfaction (Mylopoulos et al,1992), selon lesquels un but qualité est considéré « satisfait », « très satisfait », « peu satisfait » ou « pas du tout satisfait ».

La Figure 37 illustre l'évaluation qualitative de la satisfaction partielle des buts qualité. Tel que le montre la Figure 37, un but qualité complexe est décomposé jusqu'à trouver des buts qualité simples, lesquels sont satisfaits partiellement, via des *liens de contribution* (*cf.* élément Lien contribution à la Figure 37), par des buts (*cf.* élément But à la Figure 37) dont la satisfaction est précise (*i.e. hard goal*) (*cf.* Chapitre2). Ces derniers peuvent eux-mêmes être

simple ou complexe. Le but simple (*cf.* élément but simple à la Figure 37) est un but exécutable par un ensemble d'actions, tandis que le but complexe (*cf.* élément But complexe à la Figure 37) est décomposé en d'autres sous-buts, lesquels peuvent être simples ou complexes. Un but peut donc contribuer *positivement*, *très positivement*, *négativement* et *très négativement* à la satisfaction partielle des buts qualité simple.

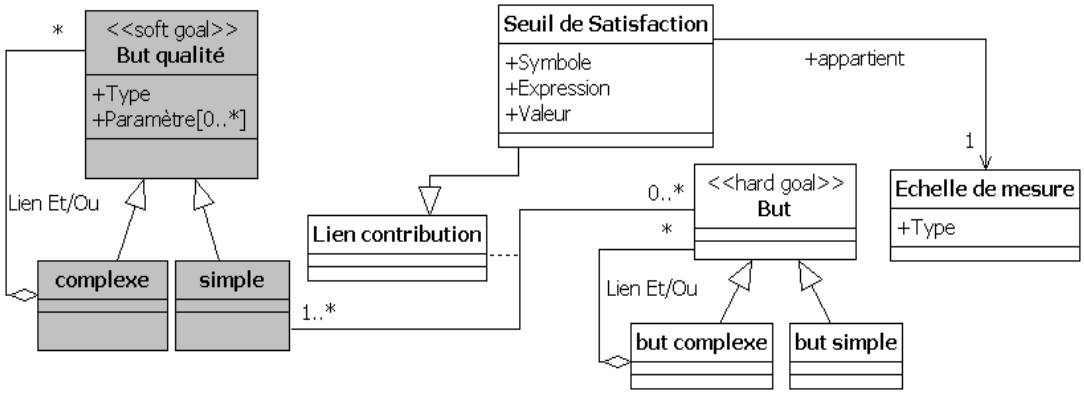

Figure 37. Satisfaction partielle

Dans un domaine particulier (*cf.* Section 5.3.2), le référentiel qualité répertorie les buts potentiels permettant de satisfaire partiellement les buts qualité. Il précise *à priori* les seuils de satisfaction pouvant être obtenu en choisissant tel but plutôt qu'un autre. Par exemple (*cf.*Figure 38), on peut considérer que les buts « contrôler le stock en continu » et « contrôler le stock par échantillonnage » contribuent, respectivement, à très bien satisfaire (« ++ ») et à ne pas satisfaire (« - ») le but qualité Fiabilité, dans le domaine de l'approvisionnement de produit.

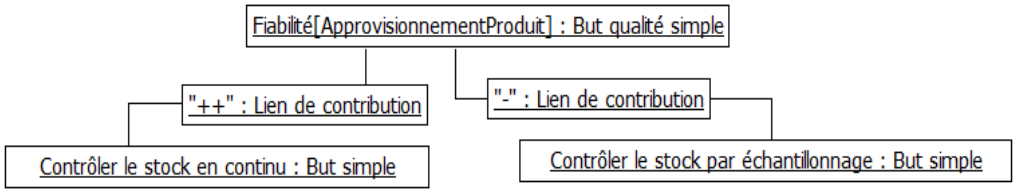

Figure 38. Exemple de la satisfaction partielle

5.3.3.1 *SEUILS DE SATISFACTION*

Il est inévitablement difficile d'utiliser une échelle absolue pour décrire la satisfaction des buts qualité, à cause de leur subjectivité et de leur relativité. Les principes de la logique floue (Guiffrida&Nagi,1998) permettent ainsi de modéliser les relations de préférence plus aisément que ne le ferait la logique booléenne. Celle-ci n'autorise pas d'affirmations de type « peu satisfait ». Les principes de la logique floue permettent également l'utilisation de

quantificateurs linguistiques (Cunha&Agard,2006), lesquels permettent de moduler la force ou l'importance de la représentation de la satisfaction. Ainsi, à l'aide de ces quantificateurs, on peut dire que les buts contribuent à « satisfaire », à « très satisfaire », à « peu satisfaire » et à « pas du tout satisfaire » les buts qualité. En effet, les quantificateurs linguistiques de force sont admis par la communauté de l'ingénierie des exigences. Nous adoptons les expressions (*cf.* attribut Expression de l'élément Seuil de satisfaction de la Figure 37) « très satisfait », « satisfait », « neutre », « peu satisfait » et « pas du tout satisfait », lesquelles sont définies comme suit :

- « très satisfait », noté par le symbole « ++ », signifie que le service contribue à un excellent niveau de satisfaction du but qualité. Il peut alors offrir une excellente satisfaction du client par rapport à cette qualité ;
- « satisfait », noté par le symbole « + », signifie que le service contribue à un bon niveau de satisfaction du but qualité. Il peut alors offrir une bonne satisfaction du client par rapport à cette qualité ;
- « neutre », noté par le symbole « ? », signifie que le service est indifférent par rapport à ce but qualité : il ne contribue ni positivement, ni négativement à la satisfaction du but qualité. Dans ce cas, on ne peut rien dire sur la satisfaction du client par rapport à cette qualité ;
- « peu satisfait », noté par le symbole « - », signifie que le service contribue à un bas niveau de satisfaction du but qualité. Il peut alors rendre le client peu satisfait par rapport à cette qualité ; et
- « pas du tout satisfait », noté par le symbole « -- », signifie que le service contribue à un niveau très bas de satisfaction du but qualité. Il peut ne pas satisfaire le client par rapport à cette qualité.

Le processus de sélection de services que nous proposons dans cette thèse (*cf.* Chapitre 8) se base, pour des raisons pratiques, sur des valeurs numériques. Par conséquent, nous proposons d'associer au seuil de satisfaction l'attribut *Valeur*. Celui-ci est une valeur numérique ($-2 \leq$ valeur $\leq +2$, valeur$\in \aleph$) permettant de traduire les seuils de satisfaction qualitatifs en des valeurs numériques, tel que présenté par (Chung&Subramanian,2001). Le Tableau 7 présente les symboles, les expressions, ainsi que les valeurs numériques (*cf.* attributs Symbole, Expression et Valeur de l'élément Seuil de satisfaction de la Figure 37) des seuils de satisfaction.

Tableau 7. Les seuils de satisfaction

Symbole	**Expression**	Valeur
++	Très satisfait	+2
+	Satisfait	+1
-	Peu satisfait	-1
--	Pas du tout satisfait	-2

5.3.3.2 *PROPAGATION DES SEUILS DE SATISFACTION*

La propagation de la satisfaction partielle des buts qualité (*i.e.* des seuils de satisfaction) à travers le graphe de décomposition consiste à calculer la satisfaction partielle des buts qualité parents. Nous reprenons les règles de propagation proposées par le cadre d'application NFR (Chung et *al.*,2000). En effet, ces auteurs définissent que dans le cas du lien d'affinement *Et*, le seuil de satisfaction du but qualité correspond au seuil minimal des buts contribuant à sa satisfaction partielle. Dans le cas du lien *Ou*, le seuil de satisfaction du but qualité correspond au seuil maximal des buts contribuant à sa satisfaction partielle.

Soient deux buts, But1 et But2, lesquels possèdent des seuils de satisfaction différents. Le Tableau 8 met en évidence la propagation de la satisfaction partielle de But1 et But2 aussi bien dans le cas d'un lien d'affinement *Et*, que d'un lien d'affinement *Ou*.

Tableau 8. Propagation de la satisfaction partielle

But1	But2	Propagation *Et*	Propagation *Ou*
++	++	++	++
	+	+	++
	-	-	++
	--	--	++
+	+	+	+
	-	-	+
	--	--	+
-	-	-	-
	--	--	-
--	--	--	--

Cette échelle qualitative ordinale (*cf.* élément Echelle de mesure à la Figure 37 sera détaillé à la Section 5.3.4.2), qui est représentée par les seuils de satisfaction, est utilisée pour deux raisons principales :

1. Tout d'abord, elle offre aux agents métier (*i.e.* les clients) un outil leur permettant d'évaluer « naturellement » leurs satisfactions. En effet, si on considère qu'un fournisseur de service publie un service de paiement, lequel utilise un protocole de cryptage particulier, le client ne maitrise pas forcément ce que ce protocole de cryptage peut lui fournir comme satisfaction. Par contre, si le fournisseur de service publie que son service de paiement peut fournir une satisfaction « + » grâce à l'utilisation de ce protocole de cryptage, alors le client comprendra plus naturellement qu'il pourra être « satisfait » par ce service.
2. Aussi, elle représente un bon moyen de communication entre les fournisseurs et les agents métier, ainsi que l'annuaire des services intentionnel, vu son caractère qualitatif. En effet, l'utilisation d'une échelle commune et suffisamment expressive (« très satisfait », « satisfait », « peu satisfait » et « pas du tout satisfait ») pour la description de la qualité de service facilite la compréhension des utilisateurs métier, et donc de la sélection du service.

Cette échelle de satisfaction est certes facile à comprendre et à utiliser par les fournisseurs et les agents du métier, mais elle reste insuffisante, pour trois raisons principales :

1. elle est très liée à la subjectivité de l'individu. En effet, deux fournisseurs de service peuvent publier un service ayant la même qualité, mais avec des seuils de satisfaction différents, car ce qui est « + » pour l'un ne l'est pas forcément pour l'autre. De façon identique, deux clients de service peuvent exiger un service de qualité identique, mais avec des seuils de satisfaction différents, car ce qui est « + » pour l'un ne l'est pas forcément pour l'autre.
2. elle peut être considérée comme injuste, vu son caractère qualitatif. En effet, deux services peuvent déclarer fournir une satisfaction « + » concernant une qualité particulière, mais, réellement, l'un d'eux peut être meilleur que l'autre concernant la même qualité. Ainsi, les fournisseurs peuvent surestimer ou sous-estimer la qualité de leurs services.
3. elle est déconnectée des valeurs de qualité fournies par les services logiciels.

Dans la perspective de réduire cette insuffisance, nous proposons, dans ce qui suit, d'associer à l'échelle qualitative une quantification des buts qualité.

5.3.4 EVALUATION QUANTITATIVE DE LA SATISFACTION DU BUT QUALITE

Dans le référentiel qualité, nous proposons d'évaluer quantitativement la satisfaction partielle du but qualité. Celle-ci se fait par des métriques, lesquelles permettent de diminuer la

subjectivité des seuils de satisfaction. Les éléments de l'évaluation quantitative sont coloriés en blanc à la Figure 31

La quantification des exigences non-fonctionnelles est présentée par Glinz (Glinz,2008) comme étant la définition des métriques permettant de rendre ces exigences mesurables. D'après Glinz (Glinz,2008), il existe des buts qualité, tel que le Temps, qui sont directement mesurables, *i.e.* auxquels des métriques sont directement associées. D'autres buts qualité ne peuvent pas être directement mesurables, tel que la Performance. Par conséquent, il faut d'abord les décomposer jusqu'à rencontrer des buts qualité mesurables et ensuite déterminer les métriques associées. Par exemple, le but qualité Performance est décomposé en Temps et Espace. Le but qualité Temps peut être mesuré par le Temps de réponse et la Rentabilité.

La Figure 39 met en évidence les éléments permettant la quantification des buts qualité dans le référentiel qualité, proposé dans cette thèse.

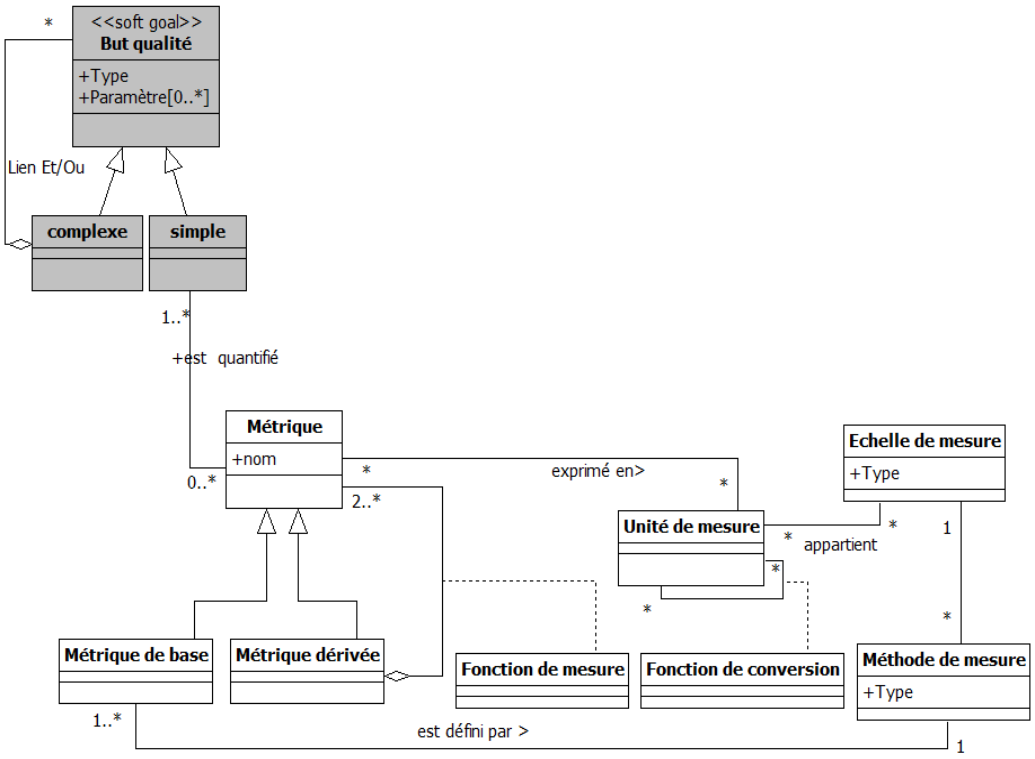

Figure 39. Définition des métriques associées à un but qualité

Comme le montre la Figure 39, les buts qualité complexes (*e.g.* Performance) sont décomposés en des buts qualité simples (*e.g.* Temps), lesquels peuvent être quantifiées par des métriques (*e.g.* Temps de réponse). Tous les buts qualité simples ne sont pas forcément quantifiables (cardinalité 0..* à la Figure 39), tel que le respect d'une règlementation particulière.

La quantification des buts qualité dans le référentiel qualité se basent sur les concepts proposés par Ruiz et ses collègues (Ruiz et *al.*,2003). Afin de fournir un consensus concernant les concepts et la terminologie à utiliser dans la mesure du logiciel, Ruiz et ses collègues (Ruiz et *al.*,2003) proposent une ontologie semi-formelle pour la mesure de logiciel. Celle-ci regroupe des concepts tels que ressource, activité, métrique, indicateur, *etc*. Dans le référentiel qualité, nous reprenons les concepts permettant d'une part d'effectuer une quantification des buts qualité, et d'autre part ces concepts peuvent être associés aux services logiciels.

Dans ce qui suit, nous proposons de définir les différents concepts de la Figure 39.

5.3.4.1 *METRIQUE*

Une *métrique* (élément Métrique à la Figure 39) est une affectation empirique d'une valeur à une entité visant à décrire une caractéristique particulière de cette entité (Xenos,2006). Dans le développement des logiciels, les métriques fournissent des indicateurs d'avancement ou de qualité de ces logiciels. Comme le stipule Ragland (Ragland,1995), les indicateurs sont des métriques ou des métriques combinées permettant de donner un aperçu sur le processus de développement du logiciel, le projet logiciel ou le logiciel lui-même.

Dans le référentiel qualité, une métrique permet de quantifier un but qualité, tels que la performance, la précision et la disponibilité. Comme définie plus haut (*cf.* Section 5.3.2), les mesures dépendent fortement du domaine d'activité considéré. Par exemple, pour mesurer la performance dans le domaine du commerce électronique on peut utiliser le « nombre de transaction/h », alors que dans celui du diagnostic *in vitro* de l'hémostase, on peut utiliser la « cadence/h ».

- La *métrique de base* (élément Métrique de base à la Figure 39) est définie par une méthode de mesure, laquelle peut être subjective ou objective ;
- La *métrique dérivée* (élément Métrique dérivée à la Figure 39) est obtenue en appliquant une fonction de mesure à au moins deux autres métriques de base ou dérivées (agrégation de cardinalité 2..* à la Figure 39). Un exemple de métrique dérivée est le calcul du « taux de fraude », lequel équivaut au nombre de fraudes signalé divisé par le nombre de transactions. Par contre, les métriques dérivées ne représentent pas de simples transformations sur les métriques de base (la racine carrée d'une métrique de base, par exemple), car leur calcul implique d'autres métriques.

Par exemple (*cf.* Figure 40), le but qualité Temps est quantifié par la métrique de base « durée d'acquisition des produits », dans le domaine de l'approvisionnement de produit.

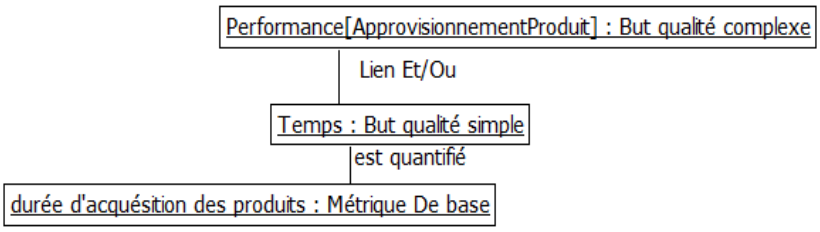

Figure 40. Exemple de métrique de base

5.3.4.2 *UNITE DE MESURE*

Chaque métrique peut avoir plusieurs *unités de mesure* (élément Unité de mesure à la Figure 39), lesquelles appartiennent à une *échelle de mesure* (élément Echelle de mesure à la Figure 39) particulière. Une échelle est un ensemble de graduations permettant de donner une fourchette de valeurs, ainsi que de quantifier des phénomènes, à travers différents signes ou manifestations extérieurs[5]. Quatre types d'échelle existent : nominal, ordinal, intervalle, et rapport.

Les échelles nominales ou qualitatives sont des échelles permettant de classer les objets dans des catégories exclusives. Ce sont des échelles non ordonnées. Par exemple, le sexe. Les échelles ordinales sont des échelles possédant les propriétés des échelles nominales, mais il existe une relation d'ordre entre les diverses catégories de l'échelle. Cette échelle s'applique aux variables quantitatives et qualitatives. Par exemple, l'échelle de la satisfaction peut s'exprimer par je suis : « très satisfait », « satisfait », « peu satisfait » ou « pas du tout satisfait », dans cet ordre.

Les échelles d'intervalle et de rapport concernent les variables quantitatives, lesquelles contiennent des valeurs mesurables. Elles sont caractérisées par des distances de taille connue entre deux valeurs quelconques de l'échelle. La différence fondamentale entre ces deux types d'échelles est liée au statut de la valeur zéro : sur l'échelle d'intervalle, le zéro est situé de manière arbitraire, comme pour la mesure des températures (par exemple, l'échelle Celsius et Fahrenheit). En revanche, sur l'échelle de rapport, le zéro a une signification précise, car il désigne l'absence du caractère considéré. Par exemple, l'âge, le poids.

[5] Wikipédia : http://fr.wikipedia.org/wiki/%C3%89chelle_(mesure)

5.3.4.3 *FONCTION DE CONVERSION*

Une *fonction de conversion* (élément Fonction de mesure à la Figure 39) est un calcul exécuté afin de passer d'une unité de mesure à une autre. Par exemple, la fonction de conversion permettant de transformer un temps *X* en secondes (s) vers un temps *Y* en minutes (min) est la suivante : *Y (min) = X(s)/60*.

5.3.4.4 *METHODE DE MESURE*

Les *méthodes de mesure* (élément Méthode de mesure à la Figure 39) peuvent être objectives ou subjectives. Les méthodes subjectives sont celles dont la quantification implique un jugement humain. Les méthodes objectives sont celles dont la quantification est basée sur des règles numériques, telles que le comptage. Ces règles peuvent être implémentées par un humain ou par un moyen automatique.

Par exemple, une méthode pour évaluer l'aisance des clients, lors de l'utilisation d'un service donné, consiste à utiliser un questionnaire en ligne. Ce type de méthode est subjectif. Dans le cas de méthode objective, la mesure peut être effectuée par un élément opérationnel (*i.e.* composant de restitution de l'information), tel que l'utilisation d'un logiciel permettant de générer des statistiques d'utilisation du service. Celles-ci permettent d'analyser le trafic engendré par le service (mesure relative de la popularité du service) et de mesurer indirectement l'aisance du client.

5.3.4.5 *FONCTION DE MESURE*

Une *fonction de mesure* (élément Fonction de mesure à la Figure 39) est un algorithme ou un calcul exécuté pour combiner deux ou plus valeurs de base et/ou dérivées. L'échelle et l'unité de la métrique dérivée dépendent des échelles et des unités des métriques de base qui la composent, ainsi que de la manière dont elles sont combinées par la fonction. Par exemple, pour la métrique dérivée « taux de fraude », définie plus haut, la fonction de mesure peut être définie comme le nombre de fraudes divisé par le nombre des transactions : taux de fraude = nombre de fraudes/nombre de transactions.

Dans le référentiel qualité, il est important de préciser aux fournisseurs d'un même domaine les unités, les échelles, les fonctions et les méthodes de mesure qu'ils peuvent utiliser pour calculer les métriques qui seront associés à leurs services logiciels. Ces éléments permettront de fournir un consensus sur la quantification des buts qualité.

5.3.5 CORRESPONDANCE ENTRE EVALUATION QUALITATIVE ET QUANTITATIVE

Chung et Subramanian (Chung&Subramanian,2001) proposent de remplacer, dans le cadre de POMSAA (*Process-Oriented Metrics for Software Architecture Adaptability*), les seuils de satisfaction par des valeurs quantitatives. Il existe plusieurs schémas de conversion vers le système métrique : (i) deux valeurs minimale et maximale (min et max) sont calculées pour les seuils ; (ii) une seule valeur est calculée pour les seuils ; ou (iii) un intervalle de valeur est calculé pour les seuils. Chung et Subramanian (Chung&Subramanian,2001) ont traité les deux premiers cas (*i.e.* deux valeurs minimale et maximale et une seule valeur).

Dans le référentiel qualité, nous proposons de traiter le troisième cas, à savoir le calcul d'un intervalle (ou d'un ensemble) de valeur pour chaque seuil de satisfaction (*cf.* Section 5.3.3.1), afin d'une part de diminuer la subjectivité de ces seuils, et d'autre part de relier les valeurs des qualités des services logiciels aux seuils de satisfaction.

Selon l'évaluation qualitative, citée préalablement (*cf.* Section 5.3.3), il est possible de préciser, par exemple, qu'un but contribue à satisfaire (« + »), à très satisfaire (« ++ »), à peu satisfaire (« - ») ou à pas du tout satisfaire (« -- ») le Temps. Cependant, pour de diminuer la subjectivité de ces seuils et d'uniformiser la sémantique des seuils de satisfaction (*i.e.* « + », « ++ », « - », « -- ») entre les différents fournisseurs et clients d'un domaine particulier, il est important de préciser des valeurs de référence. Par exemple, il s'agit de préciser que très satisfait (« ++ ») en Temps correspond à un « temps de réponse < 2 minutes ».

Ainsi, l'évaluation qualitative et l'évaluation quantitative de la satisfaction partielle du but qualité ne sont pas forcément indépendantes. Nous proposons d'établir une correspondance entre les seuils de satisfaction, utilisés dans l'évaluation qualitative, et les valeurs obtenues par les métriques dans l'évaluation quantitative. La Figure 41 montre cette correspondance entre les seuils de satisfaction et les métriques, en passant par le concept de *classe*.

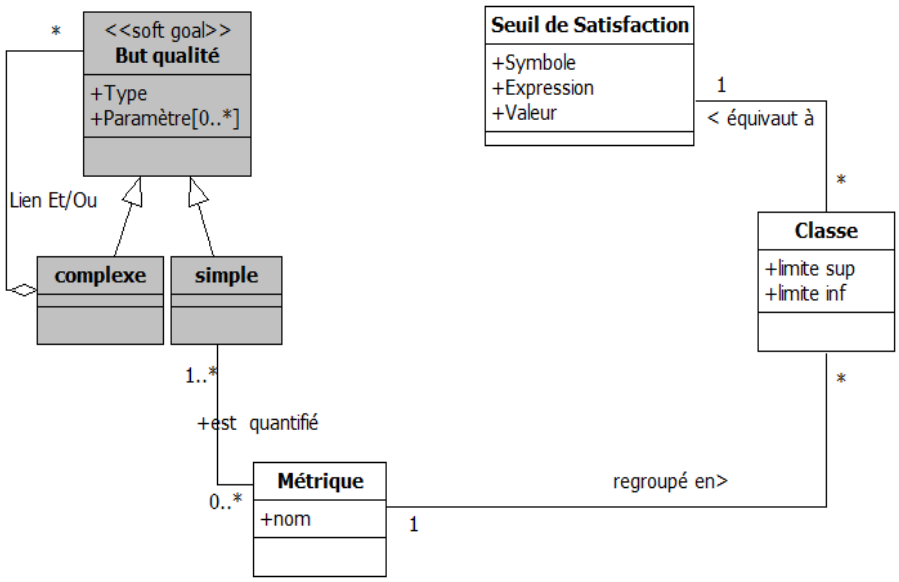

Figure 41. Lien entre les métriques et les seuils de satisfaction

Lorsque des résultats statistiques sont trop nombreux pour que les valeurs (*i.e.* les modalités en statistique) soient lisibles, il est préférable de ranger les données par intervalles appelés classes, dans le cas de valeurs continues[67].

Vu que les valeurs des métriques (*i.e.* les modalités), telles que le temps de réponse, sont nombreuses, nous proposons donc une présentation plus condensée et plus claire de ces valeurs, en les distribuant en classes disjointes, telles que chaque valeur entre dans une et une seule classe. Chaque classe est associée à un et à un seul seuil de satisfaction. Le nombre de classes est égale au nombre de seuils de satisfaction. Ce dernier appartient à une échelle de satisfaction, laquelle est exprimée par : très satisfait (« ++ »), satisfait (« + »), peu satisfait (« - ») ou pas du tout satisfait (« -- »). Le seuil « ++ » peut être associé à la classe des grandes valeurs (par exemple cadence élevée) ou à la classe des petites valeurs (par exemple, un très petit temps de réponse).

Nous proposons que le référentiel qualité s'occupe de la récupération et du classement des valeurs des métriques, en les distribuant en classes. La classe est caractérisée par deux attributs, à savoir *lim sup* et *lim inf*. L'attribut « lim sup » correspond à la limite supérieure de l'intervalle, tandis que l'attribut « lim inf » définit la limite inférieure (par exemple, lim inf < temps de réponse < lim sup). Nous proposons d'utiliser également la notion de classe pour

[6] Wikipédia : http://fr.wikipedia.org/wiki/Statistiques_%C3%A9l%C3%A9mentaires_continues#Traitement_des_donn.C3.A9es
[7] Une variable est continue si la variable prend ses valeurs dans un intervalle.

les valeurs des métriques (*i.e.* les modalités) discrètes[8]. Celle-ci (*i.e.* la classe) n'est utilisée en statistique que pour les variables continues afin d'effectuer des traitements ultérieurs, tels que la moyenne...*etc*. Dans notre cas, nous proposons de l'utiliser également pour les valeurs discrètes, car notre seul objectif est de regrouper les valeurs des métriques en classe, afin de les associer aux seuils de satisfaction. Dans ce cas, un seul attribut (« lim inf » ou « lim sup ») est utilisé pour représenter les éléments de la classe.

Par exemple (*cf.* Figure 42), la classe « lim sup = 2 jour » est associé au seuil de satisfaction « ++ ». Ceci exprime que lorsque la durée d'acquisition des produits est inférieur à 2 jours, le but qualité est considéré comme très satisfait.

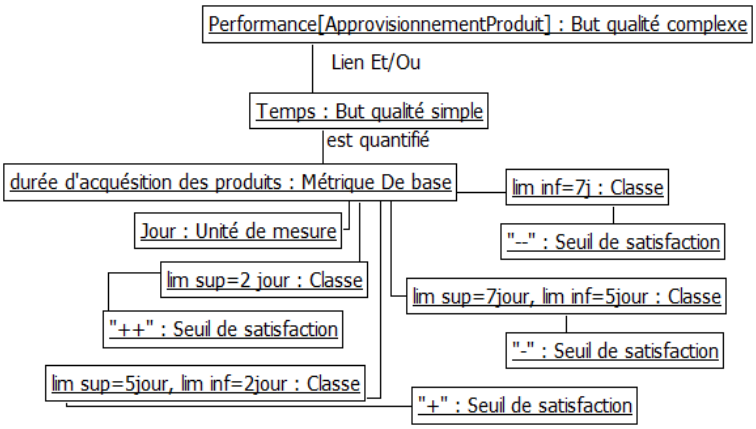

Figure 42. Exemple de métrique de base

Par exemple, le but qualité Performance peut être mesuré par la métrique « temps de réponse » (variable continue). Selon le domaine considéré, les valeurs de temps de réponse sont regroupées en classes. Le nombre de classes est égale au nombre de seuils de satisfaction. Le Tableau 9 met en évidence la correspondance entre les seuils de satisfaction et les classes de la métrique « temps de réponse », dans le cadre du but qualité Performance.

Tableau 9. Exemple de classe pour la métrique « temps de réponse »

Seuil de satisfaction	Classe
++	temps de réponse < 4mn
+	4mn <= temps de réponse < 7mn
-	7mn <= temps de réponse < 10mn
--	temps de réponse > = 10mn

Un autre exemple est le but qualité Confidentialité, lequel peut être mesuré par la métrique « protocole de cryptage » (variable discrète). Pour cette métrique, le protocole de cryptage

[8] La variable est discrète si la variable ne prends qu'un nombre fini de valeurs **,** c'est-à-dire séparées les unes des autres.

SSL (*Secure Sockets Layers*) est utilisé pour la sécurisation des transactions effectuées via Internet. Trois versions du protocole existent, à savoir SSL 1.0 ; SSL 2.0 ; et SSL 3.0. Le Tableau 10 met en évidence la correspondance entre les seuils de satisfaction et les classes de la métrique « protocole de cryptage », dans le cadre du but qualité Confidentialité.

L'intérêt de l'association des classes aux seuils de satisfaction (lien entre Classe et Seuil de satisfaction à la Figure 41) est important. En effet, la correspondance entre les seuils de satisfaction et les valeurs obtenues par les métriques permet d'associer aux seuils de satisfaction de haut niveau (« très satisfait », « satisfait », « peu satisfait » et « pas du tout satisfait ») des mesures opérationnelles, déjà fournies par les services logiciels. Même si les fournisseurs et les agents métier sont plus sensibles aux quantificateurs linguistiques de force (*i.e.* les seuils de satisfaction), les classes permettent de diminuer l'ambiguïté de ces seuils et d'uniformiser leurs sémantiques. Le référentiel qualité permet ainsi d'interpréter la qualité des services logiciels en des seuils de satisfaction facilement exploitables par les utilisateurs finaux. Aussi, le référentiel qualité permet d'améliorer la crédibilité des fournisseurs en partageant les qualités et leurs évaluations entre les différents fournisseurs.

Tableau 10. Exemple de classe pour la métrique « protocole de cryptage »

Seuil de satisfaction	Classe
++	**protocole de cryptage = SSL 3.0**
+	**protocole de cryptage = SSL 2.0**
-	**protocole de cryptage = SSL 1.0**

Cependant, le calcul des mesures opérationnelles des services logiciels n'est pas pris en compte dans cette thèse, ainsi que le processus de récupération de ces mesures. Nous proposons que le référentiel qualité s'occupe de la récupération et du classement de ces mesures en classes.

Nous avons défini dans cette section une vision détaillée du référentiel qualité en définissant le but qualité, les seuils de satisfaction et les métriques. Nous avons également présenté l'intérêt d'une double évaluation (qualitative et quantitative) de la satisfaction partielle du but qualité. Nous présentons dans la prochaine section d'instancier le méta-modèle du référentiel qualité (*cf.* Figure 31).

5.4 *INSTANCES DU META-MODELE DU REFERENTIEL QUALITE*

Nous proposons de considérer le domaine d'approvisionnement de produits (*i.e.* gestion de stock). Plusieurs buts qualité peuvent être satisfaits partiellement dans ce type de domaine

(*i.e.* le domaine d'approvisionnement de produits). Nous proposons de considérer trois buts qualité, à savoir la Fiabilité, la Performance et le Coût. Nous présentons dans ce qui suit la définition de chacun de ces buts qualité, ainsi que leurs opérationnalisations. Ces dernières, ainsi que les seuils de satisfaction, présentés dans cet exemple, dépendent simplement de notre connaissance de ce domaine.

Fiabilité : La fiabilité dans le système d'approvisionnement de produits peut concerner par exemple la fiabilité dans le contrôle de stock et la fiabilité dans le paiement des factures. Nous considérons que les buts « Contrôler le stock en continu », « Contrôler le stock périodiquement » et « Contrôler le stock par échantillonnage » contribue respectivement à la satisfaction « ++ », « + » et « - » de la Fiabilité. Le but « Arrêter en contrôlant le paiement des factures » contribue également à une satisfaction « + » de la Fiabilité. La Figure 43 présente une instance (ou un catalogue) non-exhaustive du référentiel qualité du but qualité Fiabilité concernant le domaine de l'approvisionnement de produits.

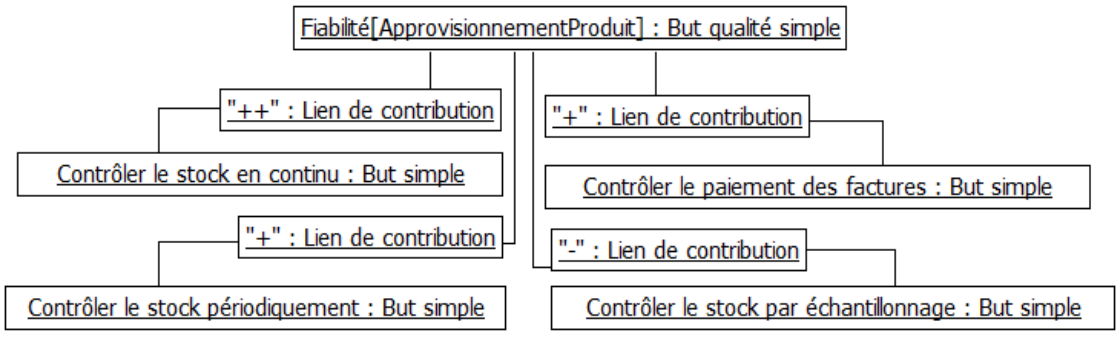

Figure 43. Le but qualité Fiabilité[ApprovisionnementProduit]

Performance : la performance dans le système d'approvisionnement de produits est définie, en termes de Temps, lequel est mesuré par la métrique « durée d'acquisition des produits », par exemple. La Figure 44 présente une instance du référentiel qualité du but qualité Performance, concernant le domaine de l'approvisionnement de produits. En effet, tel que le montre la Figure 44, la « durée d'acquisition des produits » est exprimée en jour. Quatre classes sont définies et chacune d'elles correspond à un seuil de satisfaction. Par exemple, lorsque la durée d'acquisition des produits est inférieur à deux jours, le seuil de satisfaction équivaut à « ++ ».

De plus, un ensemble de buts, permettant de satisfaire partiellement le but qualité simple Temps, dans le domaine de l'approvisionnement de produits, est recommandé. En effet, les buts « Acquérir les produits manuellement », « Acquérir les produits par seuil de

réapprovisionnement » et « Acquérir les produits par prévision stratégique » contribuent respectivement à la satisfaction « - », « + » et « ++ » à la satisfaction partielle de Temps.

Coût : le coût dans le système d'approvisionnement de produits est mesuré, par exemple par le « nombre d'employés » employé durant une période de six mois. La Figure 45 présente une instance du référentiel qualité du but qualité Coût, concernant le domaine de l'approvisionnement de produits. En effet, tel que le montre la Figure 45, la métrique « nombre d'employés » est exprimé comme un nombre entier. Quatre classes sont définies et chacune d'elles correspond à un seuil de satisfaction. Par exemple, lorsque le nombre d'employé est inférieur à deux, le seuil de satisfaction équivalent est égale à « ++ ».

De plus, un ensemble de buts, permettant de satisfaire partiellement le but qualité simple « Coût », dans le domaine de l'approvisionnement de produits, est proposé. En effet, les buts « Acquérir les produits manuellement », « Acquérir les produits par seuil de réapprovisionnement » et « Acquérir les produits par prévision stratégique » contribuent respectivement à la satisfaction « ++ », « + » et « - » de « Coût ». Les buts « Contrôler le stock en continu », « Contrôler le stock périodiquement » et « Contrôler le stock par échantillonnage » permettent également de contribuer respectivement à la satisfaction « - », « + » et « ++ » de « Coût ».

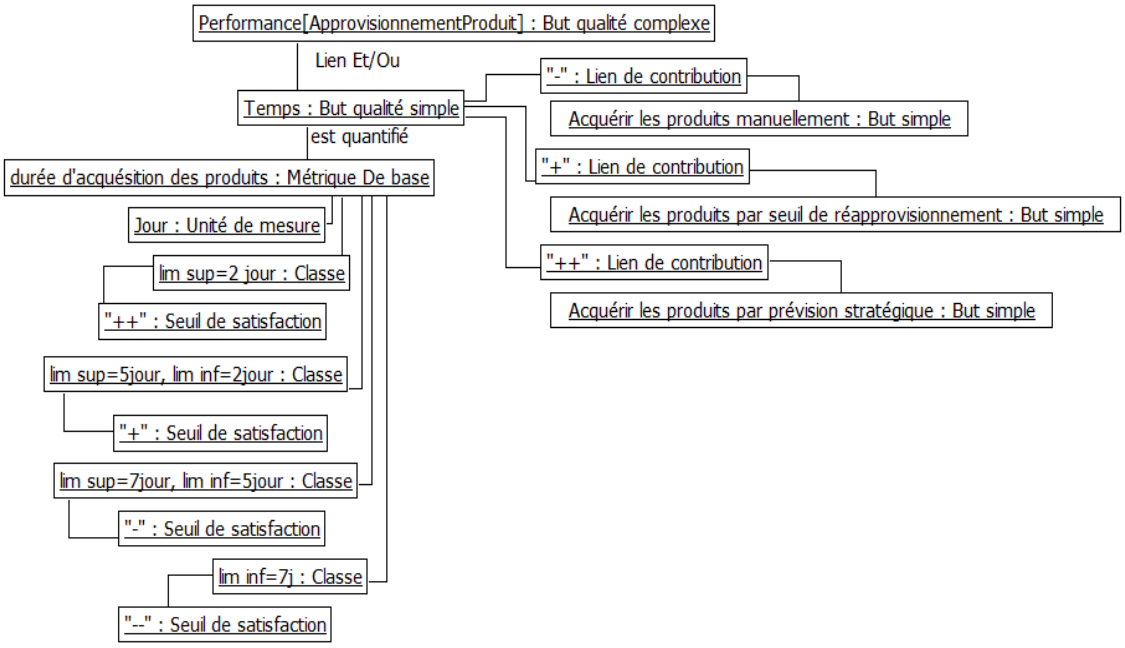

Figure 44. Le but qualité Performance[ApprovisionnementProduit]

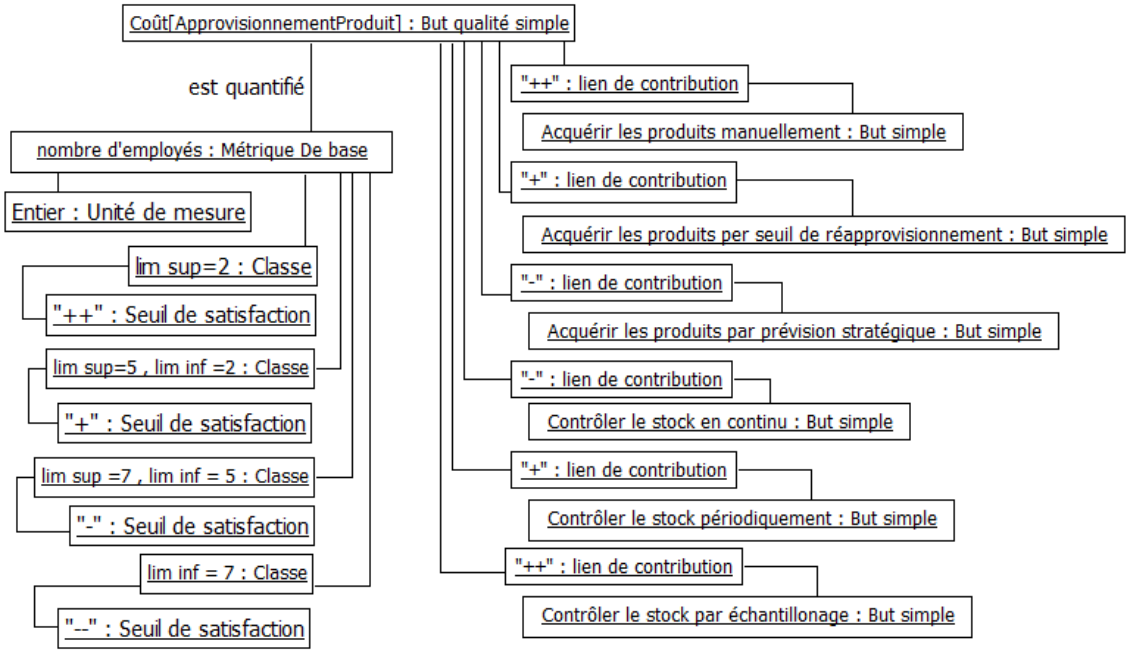

Figure 45. Le but qualité Coût[ApprovisionnementProduit]

5.5 *CONSTRUCTION DU REFERENTIEL QUALITE*

Nous proposons de définir une manière de construire le référentiel qualité. La construction d'un tel référentiel se fait par les experts d'un domaine. Par exemple, dans l'exemple du diagnostic *in vitro* de l'hémostase (*cf.* Chapitre 9), le GEHT représente les experts de ce domaine. La construction du référentiel qualité par les experts de domaine peut se faire comme suit :

Les experts de domaine collectent l'ensemble des buts (But à la Figure 31), du domaine en question, qui contribuent à satisfaire partiellement ces buts qualité. Par exemple, dans le diagnostic *in vitro* de l'hémostase, les buts « mesurer plasma par chronométrie » et « mesurer plasma par photométrie » contribuent à satisfaire partiellement la performance.

Ensuite, les experts de domaine définissent les différents buts qualité (But qualité à la Figure 31) que leur système devrait satisfaire (les buts qualité correspondent aux exigences non-fonctionnelles). Dans le domaine du diagnostic *in vitro* de l'hémostase, les buts qualité, tels que la performance de l'analyse, la confidentialité des données du patient et la précision des résultats sont préconisés.

À_la_suite, les experts de domaine définissent les métriques (Métrique à la Figure 31), lesquelles permettent de quantifier ces buts qualité. Par exemple, la Cadence est la métrique permettant de quantifier la Performance dans le domaine du diagnostic *in vitro* de l'hémostase. Les experts récupèrent les différentes valeurs des métriques, fournies par

l'exécution des conteneurs logiciels opérationnalisant les buts suscités (*i.e.* « mesurer plasma par chronométrie » et « mesurer plasma par photométrie »). Afin de relier les seuils de satisfaction à ces valeurs, les experts de domaine les rassemblent en cinq plages (Classe à la Figure 31) de valeurs. Chaque classe est alors associée à un seuil de satisfaction (Seuil de satisfaction à la Figure 31). Par exemple, la classe « 80 test/j < Cadence <=120 test/j » est associé au seuil de satisfaction « satisfait » (« + »), alors que la classe « 40 test/j < Cadence <=80 test/j » est associé au seuil de satisfaction « peu satisfait » (« - »).

Enfin, les experts de domaine définissent les contributions positives (ou très positives) et négatives (ou très négatives) entre les buts et les buts qualité. En effet, les buts, tel que « mesurer plasma par chronométrie », contribuent à satisfaire partielle les buts qualité, tel que la « performance ». La satisfaction partielle est représentée par les seuils de satisfaction, lesquels sont définis à partir des plages de valeurs. Par exemple, le but « mesurer plasma par chronométrie » contribue à satisfaire positivement (« + ») le but qualité « performance », car l'exécution du conteneur logiciel opérationnalisant ce but retourne une cadence comprise entre « 80 test/j » et « 120 test/j ».

5.6 *CONCLUSION*

Nous avons présenté dans ce chapitre le référentiel qualité, lequel fournit, par domaine, un catalogue de qualité, fournissant une homogénéisation des qualités. Il permet aux fournisseurs et aux agents métier d'utiliser les mêmes concepts et les mêmes seuils de satisfaction : les fournisseurs pour décrire la qualité de leurs services et les clients pour formaliser leurs exigences non-fonctionnelles.

Le référentiel qualité adopte les concepts du cadre d'application NFR (*NFR framework*) (Chung et *al.*,2000) pour traiter les exigences non-fonctionnelles comme des buts qualité à satisfaire partiellement. Le référentiel qualité adopte également le raisonnement de Glinz (Glinz,2008) pour faire la quantification des buts qualité. Celle-ci est effectuée en optant pour les concepts permettant d'effectuer la mesure du logiciel, proposés par Ruiz et ses collègues (Ruiz et *al.*,2003). Dans le référentiel qualité, ces concepts seront utilisés pour la mesure des services logiciels. Le référentiel qualité propose donc une double évaluation de la satisfaction partielle du but qualité : (1) l'évaluation qualitative est représentée par la contribution des buts à la satisfaction des buts qualité, via les seuils de satisfaction ; et (2) l'évaluation quantitative est représentée par la quantification des buts qualité et l'association des valeurs des métriques aux seuils de satisfaction. Cependant, cette thèse ne prend pas en compte le processus de récupération et de calcul des mesures opérationnelles des services logiciels.

Le référentiel qualité permet de diminuer l'écart existant entre la qualité délivrée par les services logiciels et les exigences non-fonctionnelles des clients. Il permet également de donner une sémantique moins subjective aux seuils de satisfaction. Il améliorer la crédibilité des fournisseurs d'un domaine, car ces derniers se basent sur le même référentiel pour décrire la qualité de leurs services. Le référentiel qualité est le catalogue de qualité réutilisé, conjointement, par les fournisseurs et les agents (ou clients) métier pour, respectivement, publier la qualité de leur service (*QoiS-Quality of intentional service*) ou exprimer leurs exigences de qualité.

La suite de ce document se base sur le référentiel qualité pour décrire la qualité du service intentionnel (*QoiS-Quality of intentional Service*) (*cf.* Chapitre 6) et pour énoncer le contexte qualité du client (*cf.* Chapitre 8).

CHAPITRE 6 : MODELE MiS-q

6.1 *INTRODUCTION*

Le service intentionnel (Kaabi,2007) est exprimé de façon à mettre en avant les buts et les stratégies que l'entreprise choisit pour les atteindre. Le service intentionnel permet de diminuer la discordance conceptuelle entre la définition des services logiciels et l'énoncé des exigences fonctionnelles des clients. L'architecture intentionnelle orientée service (*iSOA-intentional Service Oriented Architecture*) (Rolland *et al.*,2007) représente l'application de l'architecture SOA (*Service Oriented Architecture*) au niveau intentionnel. Elle propose de décrire, de publier, de rechercher et de composer les services en termes de buts et de stratégies pour les atteindre. Dans la vision iSOA, les services sont orientés but et utilisateur final, ils sont décrits dans un mode intentionnel (buts et stratégies) et publiés dans un annuaire de services intentionnels. Les clients effectuent une localisation dirigée par les buts afin de trouver les services coïncidant avec leurs exigences fonctionnelles. Cependant, le service intentionnel, tel que défini actuellement (Kaabi,2007) (Rolland *et al.*,2008), ne prend pas en compte leurs exigences non-fonctionnelles. De plus, la configuration de services agrégats dirigée par la qualité va nécessiter l'exploitation de contraintes de dépendance entre les services composant cet agrégat.

Ce chapitre présente la définition d'un modèle de représentation des contraintes de dépendance et de celle de la qualité du service intentionnel (*QoiS-Quality of intentional Service*), dénommé *MiS-q* (Modèle intentionnel de la qualité de Service). Il s'agit, en fait, d'un modèle permettant de représenter, au niveau intentionnel, les contraintes de dépendance entre services, ainsi que la qualité que peuvent offrir les services intentionnels. Les contraintes et la QoiS sont mises à la disposition des agents métier, via un mécanisme de publication.

Ce chapitre est organisé comme suit. Dans la section 2, nous proposons une vue globale du modèle *MiS-q*. La section 3 présente une vue détaillée de *MiS-q*, en présentant les contraintes de dépendance, la qualité du service intentionnel, et la spécification du service, de ses contraintes et de sa qualité, en se basant sur le langage XML (*eXtensible Markup Language*). Finalement, nous concluons ce chapitre par la section 4.

6.2 *VUE GLOBALE DU MODELE MiS-q*

La Figure 46 présente le Modèle intentionnel de la qualité de Service (*MiS-q*), en utilisant le langage UML (*Unified Modeling Language*). Le service exhibe une intentionnalité correspondant au but qu'il permet d'atteindre. Il définit les situations initiale et finale, ainsi que les pré- et post-conditions à appliquer respectivement sur la situation initiale et la situation finale pour déterminer si le but est atteint. Le modèle met également en évidence la composition de service, à savoir le service atomique et le service agrégat.

Les concepts clés du modèle *MiS-q* sont : les *contraintes de dépendance*[9] et la *QoiS*[10] (*Quality of intentional Service*).

Le service intentionnel à variation permet d'introduire plusieurs manières de réaliser le but du service. Ces différentes manières correspondent aux variantes de ce service, lesquelles font l'objet de services intentionnels. Dans la perspective de configuration de services, dirigée par la qualité, nous serons amenés à éliminer dans un service à variation, les variantes qui ne permettent pas de satisfaire le contexte qualité de l'utilisateur. Nous avons vu au Chapitre 1 et au Chapitre 3 que, dans ce cadre là, il faut vérifier que l'agrégation de service soit toujours valide, d'un point de vue métier. Le concept utilisé pour effectuer ce contrôle est une contrainte de dépendance entre variantes, que nous avons transposé en contraintes de dépendances entre services. La sélection d'une variante d'un service peut influencer la sélection d'autres services par la suite. Dans la perspective de composer (ou de configurer) le service agrégat, nous proposons de définir des contraintes de dépendance dans le modèle *MiS-q*. Ces dernières sont définies sur les variantes de service, lesquelles permettent de mettre en évidence les contraintes de type « Exige » ou « Exclut » entre les différents services intentionnels.

La QoiS du modèle *MiS-q* exhibe la qualité offerte par le service intentionnel. La QoiS peut être simple ou globale suivant, respectivement, que le service soit atomique ou agrégat. Nous définissons la QoiS globale comme une agrégation des QoiS des services composants. La QoiS simple, quant à elle, reflète la contribution des services atomiques à la satisfaction partielle (selon des seuils de satisfaction) des buts qualité. La qualité minimale, dite la $QoiS_{min}$, est également calculée, afin de préciser, aux clients de service, la qualité minimale garantie qu'un service intentionnel agrégat pourrait éventuellement fournir. La $QoiS_{min}$ est exploitée pour comparer les différents services agrégats disponibles dans l'annuaire de service

[9] Parfois désignée par dépendance, par soucis de simplicité
[10] Parfois désignée par qualité, par soucis de simplicité

(*cf.* Chapitre 8 et Chapitre 9). Dans le cas de configuration de service, nous proposons de comparer les différentes configurations du service agrégat, en calculant dynamiquement une qualité moyenne, dite $QoiS_{moy}$, chaque fois qu'une configuration exécutable est générée (*cf.* Chapitre 8 et Chapitre 9).

Les contraintes de dépendance permettent de générer des configurations exécutables de services et la qualité du service intentionnel permet de différencier aussi bien les différentes configurations propres à un même service agrégat (d'un fournisseur), que les services intentionnels publiés par divers fournisseurs et réalisant le même but. Ainsi, le service intentionnel peut répondre aux exigences non-fonctionnelles des clients, complémentaires à ses exigences fonctionnelles.

Avec le modèle *MiS-q,* nous étendons la définition et la description d'un service intentionnel de manière à considérer les contraintes de dépendance et la QoiS, lesquelles seront détaillées dans la section suivante.

6.3 *VUE DETAILLEE DU MODELE MiS-q*

Comme le présente la Figure 46, le modèle *MiS-q* met en évidence cinq aspects différents dans la description d'un service intentionnel, à savoir l'interface, le comportement, la composition, lesquels sont représentés en blanc à la Figure 46, les contraintes de dépendance entre services intentionnels, représentées en gris foncé à la Figure 46 et la qualité du service intentionnel (correspondant à la *QoiS-Quality of intentional Service*), représentée en gris clair à la Figure 46.

Les trois premiers aspects (*i.e.* l'interface, le comportement et la composition) sont décrits en détail dans le Chapitre 4. Nous proposons de détailler, dans ce chapitre, notre contribution à savoir, les aspects relatifs aussi bien aux contraintes de dépendance entre services, qu'à la qualité du service intentionnel. Nous proposons d'illustrer notre proposition (*i.e.* les concepts en gris à la Figure 46), dans ce qui suit.

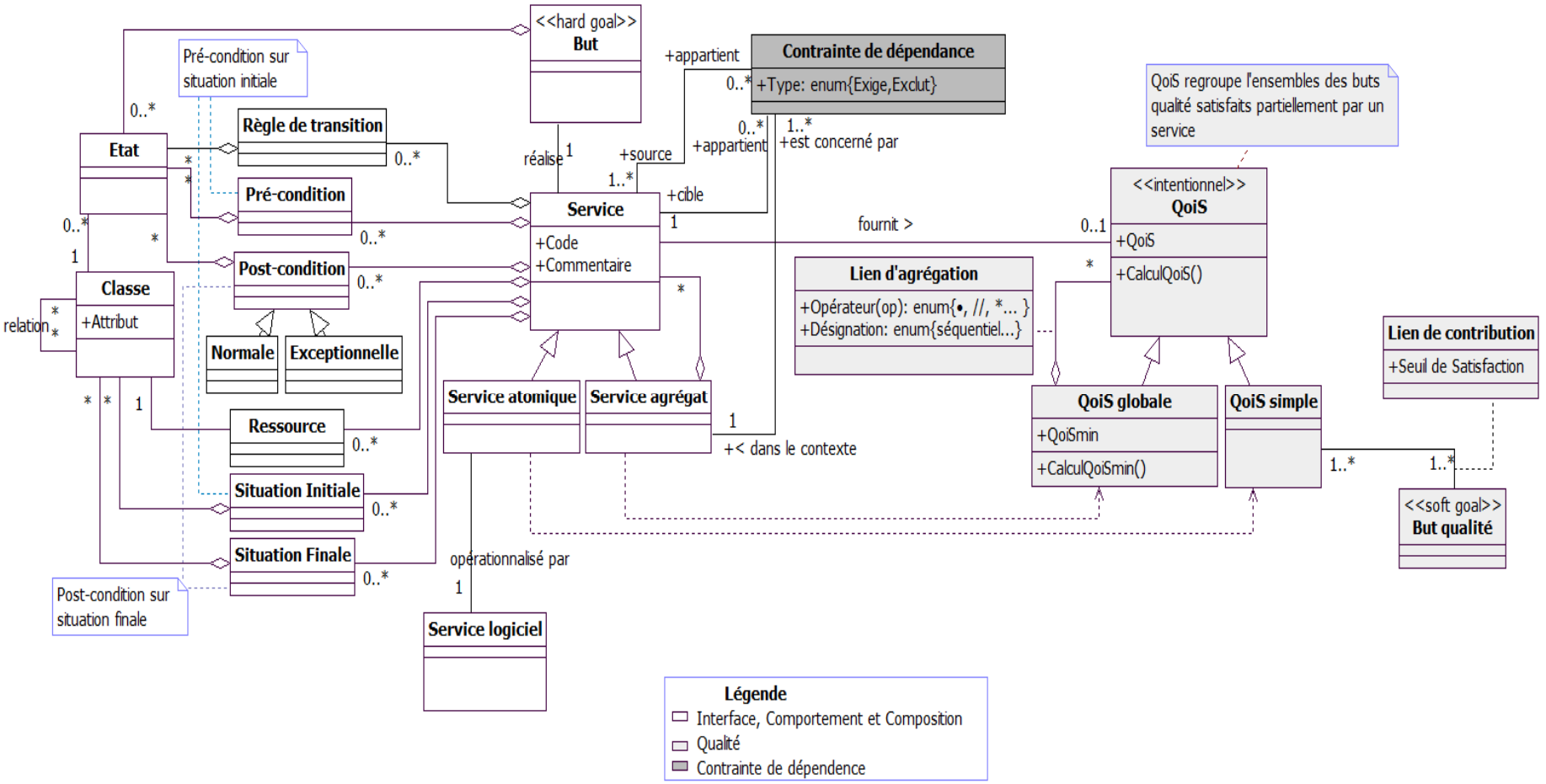

Figure 46. Le modèle *MiS-q*

6.3.1 LES CONTRAINTES DE DEPENDANCE ENTRE SERVICES INTENTIONNELS

Le service agrégat, via le service à variation, introduit de la variabilité dans la manière d'atteindre le but du service. De manière similaire à une ligne de produits logiciels, la variabilité du service intentionnel permet de mettre en évidence les caractéristiques communes à tout processus réalisant un but, tout en rendant explicite les variations possibles. Ces dernières permettent, dans le cadre d'une composition de services, de répondre aux exigences fonctionnelles des clients de différentes manières. Par contre, le fait de représenter plusieurs manières d'atteindre un but fonctionnel ne veut pas dire que toutes les combinaisons possibles sont faisables ou acceptables, que ce soit d'un point de vue technique ou métier.

Ainsi, comme le précise Bühne et ses collègues (Bühne et *al.*, 2005) pour les lignes de produit, la modélisation de la variabilité dans les exigences fonctionnelles exige la prise en compte des contraintes de dépendance afin de valider la ligne de produit. Comme ces auteurs, nous proposons, également, de définir des contraintes de dépendance entre les services intentionnels, afin de valider l'exécutabilité des configurations d'un service agrégat.

En effet, au-delà des restrictions qu'on peut indiquer à l'aide des situations initiales et finales (et des post- et pré-conditions correspondantes) de chaque service, d'autres formes de contraintes peuvent exister entre deux services. Ces contraintes puisent leurs origines aussi bien dans les aspects techniques, liés à l'opérationnalisation d'un service intentionnel par des services logiciels, que dans les aspects métier, liés au processus métier mis en place par le service intentionnel. La sélection d'une variante peut ainsi entrainer la sélection (ou l'exclusion) d'une autre variante. Par exemple, un service de transmission de données sécurisées peut exiger la mise en place d'un service de cryptographie.

Ainsi, le quatrième aspect descriptif du service intentionnel correspond aux contraintes de dépendance entre services, qui sont représentées par l'élément *Contrainte de dépendance* de la Figure 46. Nous présentons dans la suite de ce chapitre, la définition des contraintes de dépendance, l'évaluation de leur satisfaction, la définition des différents schémas de contrainte, et enfin l'ensemble des règles de validité de ces contraintes.

6.3.1.1 *DEFINITION DES CONTRAINTES DE DEPENDANCE ENTRE SERVICES*

Les langages dédiées à la modélisation de la variabilité sont essentiellement graphiques, supportés par des descriptions textuelles. Ces langages peuvent être classifiés en cinq catégories (Djebbi& Salinesi,2006) : les notations basées sur les *Features*, celles basées sur les Cas d'utilisation, celles basées sur les classes UML, celles basées sur les Buts et celles basées sur les aspects. Néanmoins, ces langages ont tous pour point commun la représentation

: (i) des éléments indispensables au développement de n'importe quel produit, *i.e.* les éléments obligatoires ; (ii) des éléments variables qui peuvent ou non être sélectionné pour faire partie du produit à développer ; et (iii) des contraintes d'assemblage des éléments variables, à respecter lors de la sélection des éléments variables. Ces contraintes, définies entre deux éléments, indiquent en l'occurrence qu'un élément est incompatible avec un autre, ou qu'un élément est dépendant d'un autre. Ces contraintes sont respectivement nommées "*excludes*" et "*requires*".

Cette thèse, traitant de la composition des services intentionnels, propose de traiter également la variabilité qui peut y exister. En effet, les éléments obligatoires et variables sont exprimés par les opérateurs d'agrégation. Dans notre approche, comme les langages de variabilité, nous proposons d'introduire les contraintes « *excludes* » et « *requires* », afin de préciser lors de la sélection des variantes du service agrégat, celles qui compatibles, ou au contraire celles qui ne sont pas incompatibles.

Dans le modèle *MiS-q*, afin de répondre de différentes manières aux utilisateurs, le service intentionnel peut encapsuler de la variabilité fonctionnelle à travers le service à variation ($\otimes$, *v*, $\cup$). Nous proposons, dans cette thèse d'étendre le modèle *MiS-q* pour considérer les contraintes d'inclusion et d'exclusion, pouvant exister entre les variantes des services intentionnels. Les contraintes de dépendance, dans le modèle *MiS-q*, expriment alors des informations complémentaires aux opérateurs d'agrégation. Le concept de contrainte de dépendance entre services est alors utilisé lorsqu'une agrégation intentionnelle de service comprend plusieurs points de variation (*i.e.* plusieurs services à variation).

Comme le montre la Figure 47, les contraintes de dépendance sont définies dans le contexte d'un service agrégat donné (cardinalité *1* du côté *dans le contexte* à la Figure 47) et permettent de valider l'exécution de ses différentes configurations (ou compositions). En effet, la contrainte de dépendance est valide dans l'agrégation de services dans laquelle elle est définie. Un même service peut appartenir à plusieurs agrégations, il peut être concerné par une contrainte de dépendance dans la première agrégation et aucune dans la deuxième. Les contraintes mettent an œuvre des services mais sont valides dans le contexte d'une agrégation. Autrement dit, une contrainte peut être significative dans une agrégation et elle est insignifiante dans une autre agrégation, c'est-à-dire que chaque agrégation possède ses propres contraintes. En effet, on associe au service agrégat une collection de contraintes de dépendance (cardinalité 1..* du côté *est concerné par* à la Figure 47), lesquelles doivent être toutes satisfaites pour qu'une configuration de ce service soit exécutable. La Figure 48 montre

que le service agrégat $\mathbf{S}_{ag}$ est concerné par trois contraintes de dépendance *C1*, *C2* et *C3*. Les configurations exécutables de $\mathbf{S}_{ag}$ sont celles qui satisfont les trois contraintes.

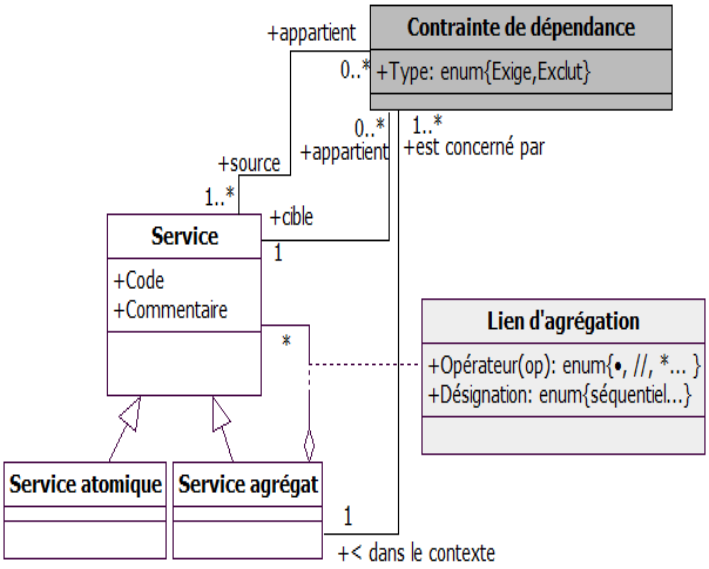

Figure 47. Le modèle de contrainte de dépendance entre services intentionnels

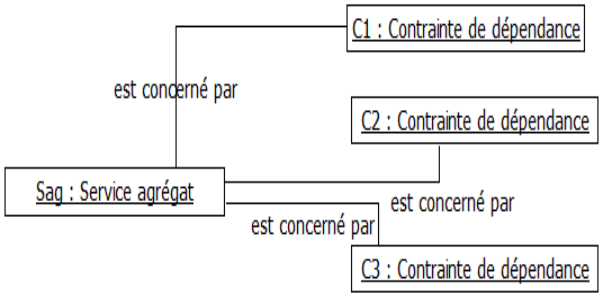

Figure 48. Instanciation de la relation « est concerné par » de la Figure 47

Par soucis de réutilisation, le service agrégat encapsule plusieurs points de variation. Les différentes variantes de ce service ne sont pas forcément toutes compatibles ou faisable. Pour cela, on a besoin de considérer, comme (Lutz,2000) (Bühne et *al.*, 2005), des contraintes de variabilité correspondant aux restrictions concernant la compatibilité de ces variantes. Pour cela, les contraintes d'inclusion et d'exclusion permettent respectivement de préciser les variantes devant obligatoirement (*i.e.* la contrainte Exige) faire partie de la même configuration, pour que celle-ci soit exécutable, et celles, au contraire, qui empêchent (*i.e.* la contrainte Exclut) l'exécution de cette configuration. La configuration du service agrégat correspond alors à effectuer de manière consécutive des choix de variantes. Ces choix sont guidés d'une part par les opérateurs d'agrégation (*cf.* Chapitre 4, pour plus de détail sur les opérateurs d'agrégation), et d'autre part par les contraintes de dépendance. Les contraintes de

dépendance, complémentaires aux opérateurs d'agrégation, concernent particulièrement les variantes du service agrégat : le choix d'une variante donnée a une influence sur le choix suivant de variantes. En effet, la présence d'une variante dans une configuration de services peut solliciter la présence d'autres variantes dans la suite (*i.e.* plus loin dans la configuration), ou au contraire en éliminer d'autres. Nous considérons alors, dans cette thèse, deux types de contraintes de dépendance : les contraintes *Exige* et *Exclut* (l'attribut Type à la Figure 47) qui seront détaillées dans la Section 6.3.1.2.

Les contraintes de dépendance sont définies entre un ou plusieurs services sources (cardinalité 1..* du coté source à la Figure 47) et un service cible (cardinalité 1 du coté cible à la Figure 47). En effet, comme le montre la Figure 47, la contrainte peut avoir comme source un service atomique ou agrégat (service composite ou à variation). Celle-ci peut également avoir comme source plusieurs services. La contrainte de dépendance possède un seul service cible (atomique ou agrégat). L'association des contraintes au service agrégat doit d'une part respecter cette définition des contraintes (*i.e.* plusieurs sources et une cible), et d'autre part la sémantique des opérateurs d'agrégation. Nous proposons alors de définir, à la Section 6.3.1.3, les règles de validité des contraintes de dépendance.

En effet, valider l'exécutabilité des configurations de service revient à vérifier que tous les services composants de ces configurations sont compatibles, *i.e.* ils peuvent s'exécuter ensemble. Autrement dit, ces configurations doivent satisfaire les contraintes de dépendance. Il s'agit alors d'évaluer la satisfaction de ces contraintes afin de valider l'exécutabilité des configurations du service agrégat. Une configuration est considérée exécutable lorsque toutes les contraintes de dépendance, associées au service agrégat, sont satisfaites. Nous proposons alors d'assimiler les contraintes de dépendance à la logique booléenne aux, et à l'évaluation de leur satisfaction. L'assimilation des contraintes à la logique booléenne est détaillée dans la Section 6.3.1.4.

6.3.1.2 *TYPOLOGIE DES CONTRAINTES DE DEPENDANCE*

Nous considérons dans cette thèse les contraintes de dépendance entre services intentionnels qui peuvent être de type *Exclut* et de type *Exige* (attribut *Type* à la Figure 47).

Comme le montre la Figure 49, la contrainte de dépendance peut être définie entre un (*cf.* Figure 49-(a)) ou plusieurs services sources (*cf.* Figure 49-(b)) et un service cible, quel que soit le type de contrainte définie (*i.e.* Exige ou Exclut). En effet, dans le contexte du service agrégat $\mathbf{S}_{ag1}$, la présence de $\mathbf{S}_1$ peut exiger ou exclure la présence de $\mathbf{S}_2$ (*cf.* Figure 49-(a)), via la contrainte de dépendance *C1*. Dans le contexte du service agrégat $\mathbf{S}_{ag2}$, la présence des

services $\mathbf{S}_1$ et $\mathbf{S}_2$ peut exiger ou exclure la présence de $\mathbf{S}_3$ (*cf.* Figure 49-(a)), grâce à la contrainte de dépendance *C2*. On peut noter que les sources d'une contrainte sont agrégées par une sémantique de conjonction.

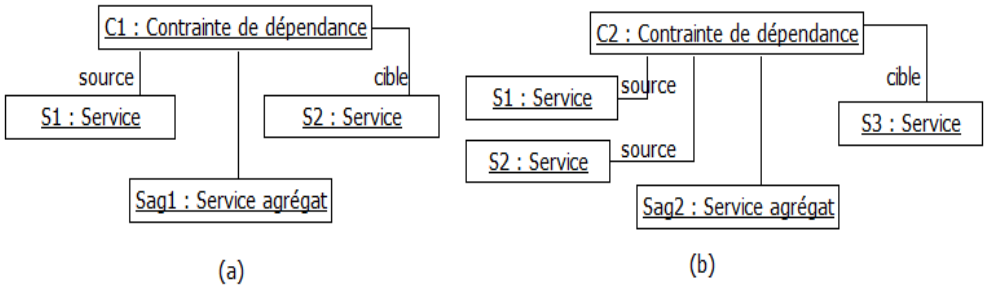

Figure 49. Instanciation de Contrainte de dépendance de la Figure 47

6.3.1.2.1 LA CONTRAINTE DE DEPENDANCE EXIGE

La contrainte de dépendance Exige exprime que la sélection d'un service nécessite la sélection d'un autre service. Elle précise une dépendance unidirectionnelle entre les services. La contrainte de dépendance Exige indique les services devant impérativement être présents dans une même configuration (ou composition) de services.

Un service $\mathbf{S}_1$ Exige un service $\mathbf{S}_2$ signifie que l'inclusion de $\mathbf{S}_1$ dans une configuration de service implique aussi l'inclusion de $\mathbf{S}_2$ dans la même configuration. La Figure 50 illustre la représentation graphique de la contrainte de dépendance Exige, dans le cas d'une seule source.

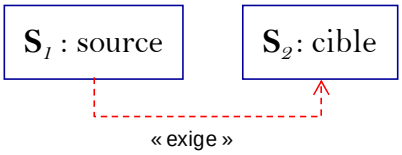

Figure 50. Représentation graphique de la contrainte de dépendance Exige avec une source

Aussi, la présence des services $\mathbf{S}_1$ et $\mathbf{S}_2$, dans une configuration donnée, peut exiger la présence du service $\mathbf{S}_3$. La Figure 51 montre la représentation graphique de la contrainte de dépendance Exige, dans le cas de deux sources.

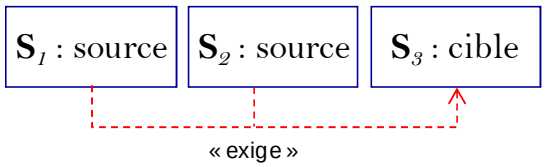

Figure 51. Représentation graphique de la contrainte de dépendance Exige avec deux sources

EXEMPLE 1

Par exemple, dans le cas d'une agrégation de service correspondant à la réservation d'un billet, nous pouvons rencontrer la contrainte d'inclusion exprimant que la réservation d'un billet à prix réduit exige la présentation d'un justificatif de réduction, d'où la contrainte d'inclusion suivante : $\mathbf{S}_{\text{Réserver un billet de train réduit}}$ Exige $\mathbf{S}_{\text{Présenter une carte de réduction}}$. Autrement dit, dans le contexte du service agrégat de réservation de billet de train, une contrainte de dépendance Exige est définie. Celle-ci possède le service $\mathbf{S}_{\text{Réserver un billet de train réduit}}$ comme source et le service $\mathbf{S}_{\text{Présenter une carte de réduction}}$ comme cible. La Figure 52 montre la représentation graphique de cette contrainte.

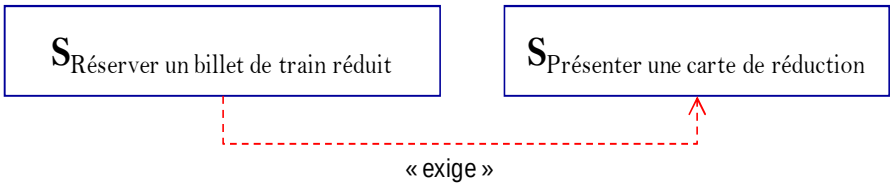

Figure 52. Représentation graphique de $\mathbf{S}_{\text{Réserver un billet de train réduit}}$ Exige $\mathbf{S}_{\text{Présenter une carte de réduction}}$

EXEMPLE 2

Toujours dans le cas de réservation d'un billet de train, les voyageurs possédant un abonnement Fréquence et souhaitant bénéficier d'une réduction du prix du billet, doivent effectuer leur réservation durant la période Bleu, d'où la contrainte d'inclusion suivante : $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Bénéficier de prix réduit}}$ Exige $\mathbf{S}_{\text{Choisir période Bleu}}$. Dans le contexte du service agrégat de réservation de billet de train, cette contrainte exprime que c'est l'exécution des deux services $\mathbf{S}_{\text{Posséder abonnement Fréquence}}$ et $\mathbf{S}_{\text{Bénéficier de prix réduit}}$ qui implique l'exécution du service $\mathbf{S}_{\text{Choisir période Bleu}}$. Autrement dit, dans cette composition, l'exécution du service $\mathbf{S}_{\text{Posséder abonnement Fréquence}}$ ou du service $\mathbf{S}_{\text{Bénéficier de prix réduit}}$ n'exige nullement l'exécution du service $\mathbf{S}_{\text{Choisir période Bleu}}$ dans la même composition. La Figure 53 montre la représentation graphique de cette contrainte.

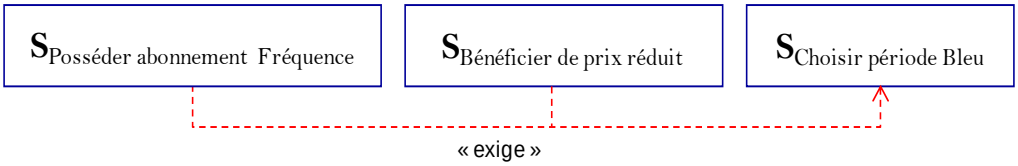

Figure 53. Représentation graphique de $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Bénéficier de prix réduit}}$ Exige $\mathbf{S}_{\text{Choisir période Bleu}}$

6.3.1.2.2 LA CONTRAINTE DE DEPENDANCE EXCLUT

La contrainte de dépendance *Exclut* exprime que la sélection d'un service écarte la sélection d'un autre service. Elle précise une dépendance bidirectionnelle entre les services. La

contrainte de dépendance Exclut indique les services ne pouvant être ensemble dans une même configuration (ou composition) de services.

Autrement dit, un service $\mathbf{S}_1$ Exclut un service $\mathbf{S}_2$ signifie que l'inclusion de $\mathbf{S}_1$ dans une configuration de service exclut $\mathbf{S}_2$ de la même configuration. La Figure 54 illustre la représentation graphique de la contrainte de dépendance Exclut, dans le cas d'une seule source.

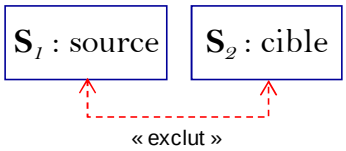

Figure 54. Représentation graphique de la contrainte de dépendance Exclut avec une source

Aussi, la présence des services $\mathbf{S}_1$ et $\mathbf{S}_2$, dans une configuration donnée, peut exclure la présence du service $\mathbf{S}_3$. La Figure 56 montre la représentation graphique de la contrainte de dépendance Exclut, dans le cas de deux sources.

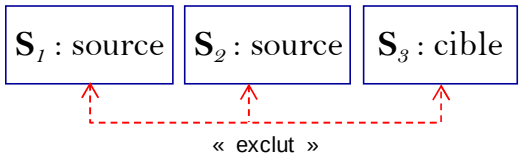

Figure 55. Représentation graphique de la contrainte de dépendance Exclut avec deux sources

EXEMPLE 1

Par exemple, dans le cas de réservation d'un billet de train par Internet, le paiement du billet en espèce est exclu, d'où la contrainte d'exclusion suivante : $\mathbf{S}_{\text{Réserverr un billet de train par internet}}$ Exclut $\mathbf{S}_{\text{Payer un billet de train par espèce}}$. Autrement dit, dans le contexte du service agrégat de réservation de billet de train, une contrainte de dépendance Exclut est définie. Cette contrainte comprend le service $\mathbf{S}_{\text{Réserverr un billet de train par internet}}$ comme source et le service $\mathbf{S}_{\text{Payer un billet de train par espèce}}$ comme cible. La Figure 56 montre la représentation graphique de cette contrainte d'exclusion.

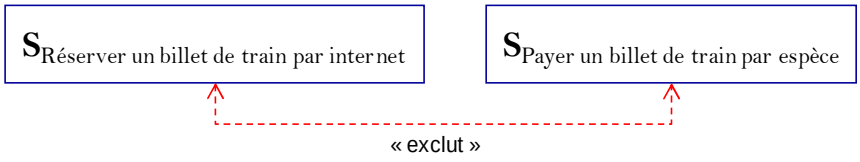

Figure 56. Représentation graphique de $\mathbf{S}_{\text{Réserverr un billet de train par internet}}$ Exclut $\mathbf{S}_{\text{Payer un billet de train par espèce}}$

EXEMPLE 2

On continue avec l'exemple de réservation d'un billet de train. Les voyageurs, qui possèdent un abonnement Fréquence, ne peuvent bénéficier d'une réduction du prix du billet lorsqu'ils réservent durant la période Rouge, d'où la contrainte d'exclusion suivante : $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Choisir période Rouge}}$ Exclut $\mathbf{S}_{\text{Bénéficier de prix réduit}}$. Autrement dit, dans le contexte de cette composition, cette contrainte d'exclusion précise que l'exécution des services $\mathbf{S}_{\text{Posséder abonnement Fréquence}}$ et $\mathbf{S}_{\text{Choisir période Rouge}}$ empêche l'exécution du service $\mathbf{S}_{\text{Bénéficier de prix réduit}}$, c'est-à-dire, qu'on ne pourra pas exécuter le service $\mathbf{S}_{\text{Bénéficier de prix réduit}}$ si on a choisit d'exécuter le service $\mathbf{S}_{\text{Posséder abonnement Fréquence}}$ et le service $\mathbf{S}_{\text{Choisir période Rouge}}$ dans la composition. La Figure 57 montre la représentation graphique de cette contrainte.

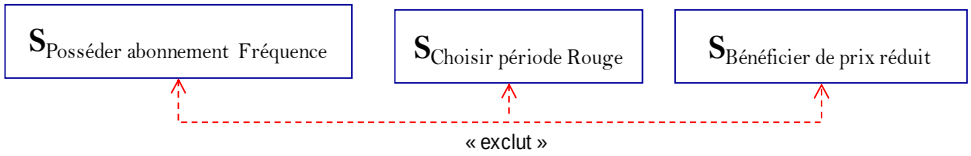

Figure 57. Représentation graphique de $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Choisir période Rouge}}$ Exclut $\mathbf{S}_{\text{Bénéficier de prix réduit}}$

6.3.1.3 *REGLES DE VALIDITE DES CONTRAINTES DE DEPENDANCE*

L'association des contraintes de dépendance au service agrégat doit respecter d'une part la définition des contraintes, présentée plus haut (*cf.* Section 6.3.1.1), et d'autre part elle ne doit pas transgresser la sémantique des opérateurs d'agrégation.

Afin de valider les contraintes de dépendance définies, nous proposons, dans ce qui suit, des règles de validité des contraintes. Ces règles doivent être respectées lors de la définition des contraintes de dépendance. Nous tenons à préciser qu'un ensemble de contraintes de dépendance est associé au service agrégat. Cet ensemble exprime une conjonction de contraintes définissant que toutes ces contraintes doivent être satisfaites pour qu'une configuration de ce service soit exécutable.

Dans le cas de contrainte de plusieurs services en cible de type « S_1 exige (ou exclut) S_2 Et S_3 », nous proposons d'appliquer R1.

R1 : si la cible est composée de plusieurs services, il faut décomposer celle-ci en plusieurs contraintes avec pour chacune d'entre elles une seule cible

Par exemple, on peut définir une contrainte de ce type : $\mathbf{S}_1$ exige $\mathbf{S}_2$ et $\mathbf{S}_3$ (*cf.* Figure 58-(a)). Cette contrainte exprime qu'une configuration qui inclut le service $\mathbf{S}_1$ doit inclure aussi les services $\mathbf{S}_2$ et $\mathbf{S}_3$. En appliquant R1, cette contrainte est alors transformée en : ($\mathbf{S}_1$ exige $\mathbf{S}_2$) et

($\mathbf{S}_1$ exige $\mathbf{S}_3$), car comme défini plus haut, nous considérons qu'une configuration est exécutable lorsqu'elle satisfait *toutes* les contraintes associées au service agrégat (conjonction de contraintes). De façon identique, $\mathbf{S}_1$ exclut $\mathbf{S}_2$ et $\mathbf{S}_3$ (*cf.* Figure 58

Figure 59-(b)) est transformée en : ($\mathbf{S}_1$ exclut $\mathbf{S}_2$) et ($\mathbf{S}_1$ exclut $\mathbf{S}_3$).

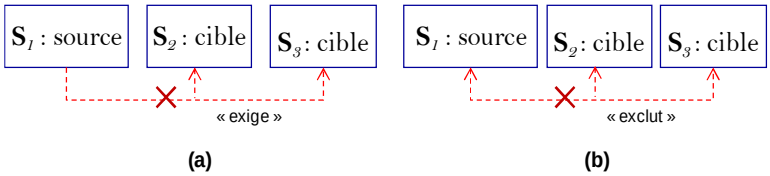

Figure 58. Les contraintes interdites dans le cas de plusieurs services en cible

Dans le cas d'un service à choix alternatif (⊗) ou d'un service à variation de chemin (∪), nous proposons d'appliquer R2.

R2 : pas de contrainte de dépendance entre les variantes (de manière directe ou indirecte) d'un service à choix alternatif (⊗) et celles d'un service à variation de chemin (∪).

En effet, le service à choix alternatif (⊗) et le service à variation de chemin (∪) représentent une relation d'exclusion (*i.e.* relation de type *Ou Exclusif*) entre leurs variantes. Cette relation exprime que le choix d'une variante empêche la sélection des autres variantes. La contrainte d'exclusion entre les variantes est alors déjà exprimée par les opérateurs ⊗ et ∪. Il est alors inutile de définir une contrainte d'exclusion entre les variantes de ces services (*cf.* Figure 59-(b) et Figure 60-(b)). De plus, il ne peut exister de contrainte Exige entre les variantes, car, par définition, toutes les variantes s'excluent mutuellement (*cf.* Figure 59-(a) et Figure 60-(a)). Si des variantes requièrent d'autres, alors il existe forcément une erreur de conception du service en question et une mauvaise utilisation des opérateurs ⊗ et ∪.

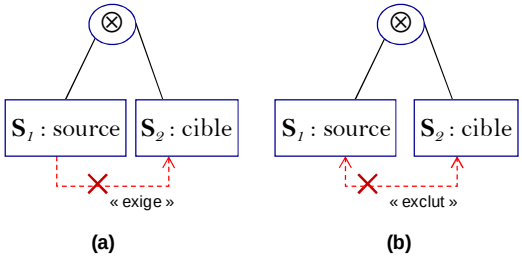

Figure 59. Les contraintes interdites dans le cas d'un service à choix alternatif

Par exemple, le service de paiement d'un billet de train est défini comme suit : $\mathbf{S}_{\text{Payer billet de train}} = \otimes(\mathbf{S}_{\text{Payer billet par carte bancaire}}, \mathbf{S}_{\text{Payer billet par chèque}}, \mathbf{S}_{\text{Payer billet par espèce}})$. Autrement dit, le paiement se fait exclusivement par carte bancaire, par chèque ou par espèce. On ne peut définir une contrainte d'exclusion, car celle-ci est indiquée implicitement par le service lui même. Aussi,

on ne peut associer aux variantes de ce service une contrainte d'inclusion, car on ne peut pas cumuler les modes de paiement. Si c'est le cas, alors il existe une erreur de conception du service $\mathbf{S}_{\text{Payer billet de train}}$.

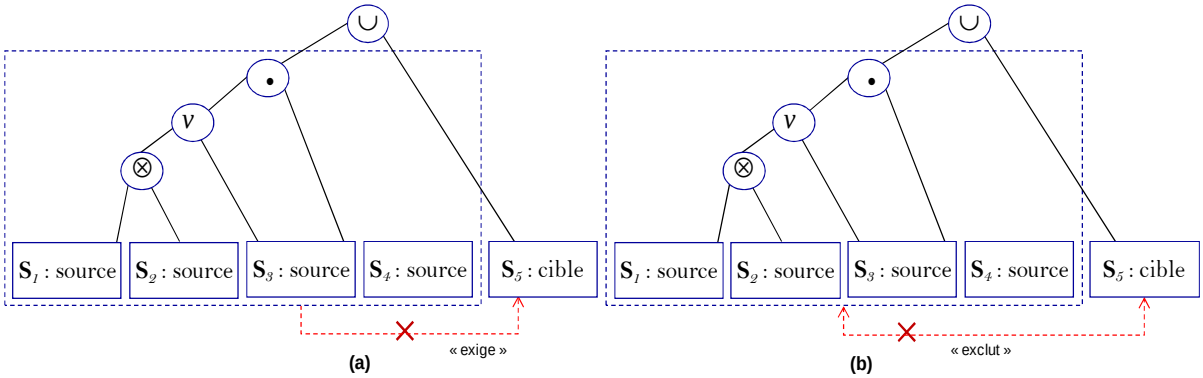

Figure 60. Les contraintes interdites dans le cas d'un service à variation de chemin

Dans le cas d'un service à choix multiple (*v*), nous proposons d'appliquer R3.

R3 : pas de contrainte de dépendance entre les variantes d'un service à choix multiple (v).

Le service à choix multiple (*v*) exprime un choix non-exclusif entre les services composants, ce qui correspond à un affinement *Ou*. Il regroupe plusieurs services composants parmi lesquels au moins un, mais éventuellement plusieurs, peuvent être choisis (*cf.* Chapitre 4). Le fait de pouvoir choisir plusieurs variantes de ce service implique que nous ne pouvons pas définir de contrainte d'inclusion entre ces variantes, car celle-ci est définie implicitement par le comportement du service à choix multiple (*v*) (*cf.* Figure 61-(a)). Aussi, la relation *Ou* représente l'alternative entre les variantes de ce service, sans autant qu'elle soit exclusive. Néanmoins, on ne peut définir une contrainte d'exclusion entre ces variantes (*cf.* Figure 61-(b)), car si un besoin d'exclusion entre les variantes d'un service à choix multiple se fait ressentir, il faut alors introduire l'opérateur ($\otimes$), pour répondre à ce besoin.

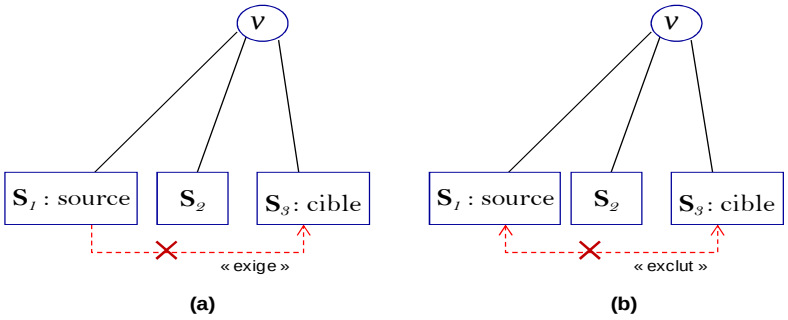

Figure 61. Les contraintes interdites dans le cas d'un service à choix multiple

Par exemple, si un client souhaite effectuer une recherche d'un billet de train, il peut le faire par date, par destination ou par prix. Le service de recherche d'un billet de train est défini

comme suit : $\mathbf{S}_{\text{Rechercher billet de train}} = \nu\ (\mathbf{S}_{\text{Rechercher billet par date}}, \mathbf{S}_{\text{Rechercher billet par destination}}, \mathbf{S}_{\text{Rechercher billet par prix}})$. Ce service exprime qu'un ou plusieurs services peuvent être utilisés pour rechercher un billet de train. L'exécution de $\mathbf{S}_{\text{Rechercher billet par date}}$ ou de $\mathbf{S}_{\text{Rechercher billet par destination}}$ ou de $\mathbf{S}_{\text{Rechercher billet par prix}}$ peut suffire pour atteindre le but du service $\mathbf{S}_{\text{Rechercher billet de train}}$. Aussi, il est possible d'exécuter les trois services pour réaliser le but du même service. La contrainte d'inclusion est alors inutile. La contrainte d'exclusion est également inutile, autrement cela revient à dire, par exemple, que l'exécution de $\mathbf{S}_{\text{Rechercher billet par date}}$ empêche l'exécution de $\mathbf{S}_{\text{Rechercher billet par destination}}$ et de $\mathbf{S}_{\text{Rechercher billet par prix}}$.

Dans le cas d'un service séquentiel *(•) (ou parallèle (//)*, nous proposons d'appliquer R4.

R4 : pas de contrainte de dépendance entre les services composants d'un service séquentiel (•) (ou parallèle (//)).

Le service séquentiel (•) (ou parallèle (//)) représente une relation de succession entre ses services composants. Les contraintes de dépendance ne peuvent être définies entre les services composants d'un service séquentiel, car la relation de succession indique qu'un service composant ne peut être exécuté que si un autre a été exécuté. Elle permet d'ordonner les services composants dans une configuration. Il ne peut alors pas exister de contrainte d'inclusion entre ces services composants, car celle-ci est définie implicitement par l'opérateur d'agrégation (•) (*cf.* Figure 62-(a)). Aussi, il ne peut exister de contrainte d'exclusion, car tous les services composants sont obligatoires dans le service séquentiel (*cf.* Figure 62-(b)). S'il existe des services composants qui s'excluent mutuellement, il existe alors inévitablement une erreur de conception du service en question.

Par exemple, si un client désire réserver un billet de train, il s'agit d'abord de rechercher le billet (*i.e.* $\mathbf{S}_{\text{Rechercher billet de train}}$), de payer le billet (*i.e.* $\mathbf{S}_{\text{Payer billet de train}}$), et enfin d'imprimer le billet de train (*i.e.* $\mathbf{S}_{\text{Imprimer billet de train}}$). Ce service reflète l'exécution d'une succession de services pour réserver un billet de train. On ne peut définir de contrainte d'inclusion entre les services composants de ce service, car la sémantique du service séquentiel implique que tous les services composants doivent être exécutés. La contrainte d'inclusion est donc inutile. Aussi il ne doit pas avoir de contrainte d'exclusion, sinon il y a une erreur de conception du service de réservation de billet de train.

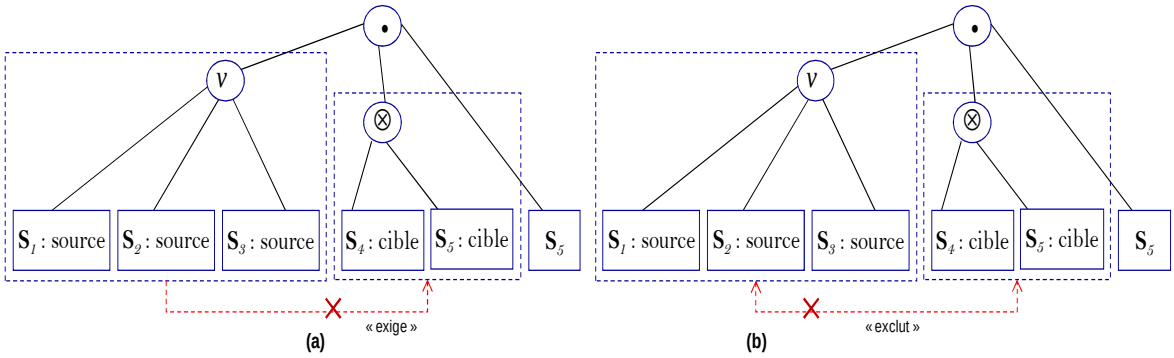

Figure 62. Les contraintes interdites dans le cas d'un service séquentiel

6.3.1.4 *EVALUATION DES CONTRAINTES DE DEPENDANCE*

La majorité des approches de lignes de produit (Batory, 2005) (Djebbi, 2011) modélisent les contraintes de dépendance par des expressions booléennes[11], correspondant aux formules propositionnelles[12] (Tarski,2008).

Dans ce qui suit, nous présentons d'abord les concepts de la logique booléenne. Ensuite, il faut valider l'ensemble des contraintes d'un service agrégat, en utilisant la logique booléenne. Celle-ci est utilisée d'une part pour exprimer les contraintes de dépendance de *MiS-q*, et d'autre part pour vérifier la cohérence des configurations.

6.3.1.4.1 LA LOGIQUE BOOLEENNE

Les expressions booléennes (Tarski,2008) sont définies par :

- un ensemble de valeurs de vérité E= {Vrai, Faux}. Cet ensemble est aussi noté : E = {1, 0}.

Dans notre cas, nous considérons que les valeurs « 1 » et « 0 » correspondent, respectivement, à la participation (*i.e.* « 1 ») ou non (*i.e.* « 0 ») d'un service intentionnel à une configuration de services donnée.

- des variables propositionnelles a, b, … à valeurs dans E.

Dans notre cas, nous considérons que les variables propositionnelles correspondent aux différents services $\mathbf{S}_1$, $\mathbf{S}_2$… à valeurs dans E. En effet, si $\boldsymbol{S_1=1}$ alors le service $\boldsymbol{S_1}$ fait partie de la configuration de services, alors que dans le cas où $\boldsymbol{S_1=0}$ le service $\boldsymbol{S_1}$ n'en fait pas partie.

- des opérateurs logiques permettent de relier les différentes variables, tels que la conjonction, la négation et l'implication.

[11] Une expression booléenne est une expression qui a pour valeur le type de données booléen.
[12] C'est-à-dire formées à partir des variables booléennes et des opérateurs logiques.

- La conjonction correspond à un Et logique entre les variables. Elle est notée par l'opérateur : « $\wedge$ ». La conjonction est définie de la manière suivante : $(a \wedge b)=1$ si et seulement si $(a=1)$ et $(b=1)$.
- La négation (ou le contraire) est notée par l'opérateur « $\neg$ ». La négation est définie de la manière suivante : $\neg (a)=1$ si et seulement si $(a=0)$.
- L'implication est notée par l'opérateur : « $\Rightarrow$ ». L'implication $a \Rightarrow b$ correspond à l'expression $(\neg (a) \vee b)$. La variable *a* est une condition suffisante pour la variable *b*, qui, elle, est une condition nécessaire pour *a*. Autrement dit, tant que $a = 0$ alors $(a \Rightarrow b) = 1$ quelque soit la valeur de *b* (la valeur de b est ignorée). Si $a = 1$ alors $(a \Rightarrow b)$ prendra une valeur *0* ou *1* en fonction de la valeur de *b* ; *b* est alors impliqué dans le résultat de la valeur de $(a \Rightarrow b)$.

Le Tableau 11 représente un récapitulatif des opérateurs logiques cités précédemment.

Tableau 11. Récapitulatif des opérateurs logiques

a	b	$\neg a$	$a \wedge b$	$a \Rightarrow b$
0	0	1	0	1
0	1	1	0	1
1	0	0	0	0
1	1	0	1	1

6.3.1.4.2 ASSIMILATION DES CONTRAINTES DE DEPENDANCE A L'ALGEBRE BOOLEENNE

Comme précisé plus haut (*cf.* Figure 47), les contraintes de dépendance sont définies dans le contexte d'un service agrégat $\mathbf{S}_{ag}$, et elles concernent les variantes de ce service $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{S}_3$, *etc*. Ces dernières peuvent être des services atomiques ou des services agrégats. Une contrainte de dépendance est déterminée par une type de contrainte (exige ou exclut), un ou plusieurs services sources et un service cible (*cf.* Figure 47).

Nous proposons de formaliser la définition des contraintes de dépendance à l'aide de la logique booléenne et plus précisément des opérateurs de négation et d'implication.

6.3.1.4.2.1 *ASSIMILATION DE LA CONTRAINTE DE DEPENDANCE EXIGE A L'ALGEBRE BOOLEENNE*

Nous proposons d'assimiler la contrainte de dépendance Exige à l'opérateur d'implication ($\Rightarrow$). L'usage de l'opérateur d'implication nous permet ainsi d'exprimer la contrainte de dépendance unilatérale qui relie le service $\mathbf{S}_1$ au service $\mathbf{S}_2$: si $\mathbf{S}_1$ est inclus dans une configuration, alors le service $\mathbf{S}_2$ doit aussi l'être pour que la contrainte d'inclusion soit satisfaite. La contrainte de dépendance Exige peut alors s'exprimer par :

$$\mathbf{S}_1 \Rightarrow \mathbf{S}_2$$

EXEMPLE 1

Dans le contexte du service agrégat de réservation de billet de train, l'achat d'un billet de train à prix réduit exige du voyageur de présenter, le jour de son départ, sa carte de réduction. La contrainte de dépendance correspondant à cet exemple est alors exprimée comme suit : $\mathbf{S}_{\text{Réserver un billet de train réduit}} \Rightarrow (\mathbf{S}_{\text{Présenter une carte de réduction}})$.

EXEMPLE 2

Dans le contexte du même service agrégat, les voyageurs possédant un abonnement Fréquence et souhaitant bénéficier d'une réduction du prix du billet, doivent effectuer leur réservation durant la période Bleu. La contrainte de dépendance correspondant à cet exemple est alors exprimée par : $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Bénéficier de prix réduit}} \Rightarrow (\mathbf{S}_{\text{Choisir période Bleu}})$.

6.3.1.4.2.2 ***ASSIMILATION DE LA CONTRAINTE DE DEPENDANCE EXCLUT A L'ALGEBRE BOOLEENNE***

Nous proposons d'assimiler la contrainte de dépendance Exclut à l'opérateur de négation ($\neg$), associé à l'opérateur ($\Rightarrow$). L'opérateur de négation ($\neg$) nous permet ainsi d'exprimer le fait que, si le service $\mathbf{S}_1$ est inclus dans une configuration, le service $\mathbf{S}_2$ ne devrait pas l'être : il doit être absent de la même configuration. La contrainte de dépendance Exclut peut s'exprimer comme suit :

$$\mathbf{S}_1 \Rightarrow \neg(\mathbf{S}_2)$$

EXEMPLE 1

Dans le contexte du service agrégat de réservation de billet de train, le paiement d'un billet de train en espèce est interdit lorsque les voyageurs ont réservé leurs billets sur internet. La contrainte de dépendance correspondant à cet exemple est alors exprimée comme suit : $\mathbf{S}_{\text{Réserver un billet de train par internet}} \Rightarrow \neg(\mathbf{S}_{\text{Payer un billet de train par espèce}})$.

EXEMPLE 2

Dans le contexte du même service agrégat, les voyageurs, qui possèdent un abonnement Fréquence, ne peuvent bénéficier d'une réduction du prix du billet lorsqu'ils réservent durant la période Rouge. La contrainte de dépendance correspondant à cet exemple est alors exprimée par : $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Choisir période Rouge}} \Rightarrow \neg(\mathbf{S}_{\text{Bénéficier de prix réduit}})$.

6.3.1.4.3 L'EVALUATION BOOLEENNE DES CONTRAINTES DE DEPENDANCE

Plusieurs contraintes de dépendance peuvent être associées à un service agrégat. Elles permettent de valider l'exécutabilité des configurations de ce service. Une configuration de service est considérée exécutable lorsque l'ensemble des contraintes de dépendance, associées au service agrégat, sont satisfaites. Nous considérons cet ensemble de contraintes comme

étant une conjonction de contraintes exprimant que toutes les contraintes doivent être satisfaites pour qu'une configuration soit exécutable.

Afin d'évaluer la satisfaction des contraintes de dépendance, nous considérons que si le service $\mathbf{S}_1$=1, alors $\mathbf{S}_1$ est inclue à la configuration de services. Inversement, si $\mathbf{S}_1$=0 alors le service $\mathbf{S}_1$ ne fait pas partie de la configuration. L'évaluation booléenne des contraintes de *MiS-q* est exploitée lors du processus de configuration de service (*cf.* Chapitre 8) et elle est implémentée dans le prototype TOPSiCQ (*cf.* Chapitre 9), afin de valider l'exécutabilité des configurations du service agrégat.

6.3.1.4.3.1 *L'EVALUATION DE LA CONTRAINTE DE DEPENDANCE EXIGE*

Le Tableau 12 résume l'évaluation booléenne de la contrainte de dépendance Exige. Comme le montre le Tableau 12, la contrainte d'exigence est satisfaite (*i.e.* $(\mathbf{S}_1 \Rightarrow \mathbf{S}_2)$=1) dans les cas suivants :

- lorsque le service $\mathbf{S}_1$ est inclus à la configuration de service ($\mathbf{S}_1$=1 à la ligne 4 du Tableau 12), alors le service $\mathbf{S}_2$ doit également l'être ($\mathbf{S}_2$=1 à la ligne 4 du Tableau 12), pour satisfaire la contrainte d'exigence.
- lorsque le service $\mathbf{S}_1$ ne fait pas partie de la configuration de service ($\mathbf{S}_1$=0 à la ligne 1 et ligne 2 du Tableau 12), la contrainte est satisfaite quelle que soit la présence (ou non) du service $\mathbf{S}_2$ ($\mathbf{S}_2$=1 à la ligne 2, $\mathbf{S}_2$=0 à la ligne 1 du Tableau 12).

Si le service $\mathbf{S}_1$ fait partie de la configuration ($\mathbf{S}_1$=1 à la ligne 3 du Tableau 12) et le service $\mathbf{S}_2$ ne fait pas partie de la configuration ($\mathbf{S}_2$=0 à la ligne 3 du Tableau 12), alors la contrainte de dépendance n'est pas satisfaite (($\mathbf{S}_1 \Rightarrow \mathbf{S}_2$)=0).

Autrement dit, lorsque le service $\mathbf{S}_1$ ne fait partie de la configuration, on a pas besoin d'évaluer $\mathbf{S}_2$. Contrairement, si le service $\mathbf{S}_1$ est présent dans la configuration, il est impératif d'évaluer $\mathbf{S}_2$, qui doit être présent dans cette configuration pour que la contrainte soit satisfaite.

Tableau 12. Evaluation de la contrainte de dépendance Exige

Ligne	$\mathbf{S}_1$	$\mathbf{S}_2$	$S_1 \Rightarrow S_2$
1	0	0	**1**
2	0	1	**1**
3	1	0	0
4	1	1	**1**

EXEMPLE

Le Tableau 13 résume l'évaluation de la contrainte de dépendance $\mathbf{S}_{\text{Réserver un billet de train réduit}} \Rightarrow (\mathbf{S}_{\text{Présenter une carte de réduction}})$. Le Tableau 13 montre que la présence du service $\mathbf{S}_{\text{ Réserver un billet de}}$

train réduit dans la configuration de service sollicite la présence du service $\mathbf{S}_{\text{Présenter une carte de réduction}}$, pour que la contrainte Exige soit satisfaite (ligne 3 au Tableau 13). Aussi, cette contrainte est satisfaite lorsque le $\mathbf{S}_{\text{Réserver un billet de train réduit}}$ n'est pas présent dans la configuration (ligne 1 et ligne 2 au Tableau 13).

Tableau 13. Evaluation de $\mathbf{S}_{\text{Réserver un billet de train réduit}} \Rightarrow (\mathbf{S}_{\text{Présenter une carte de réduction}})$

Ligne	$\mathbf{S}_{\text{Réserver un billet de train réduit}}$	$\mathbf{S}_{\text{Présenter une carte de réduction}}$	$\mathbf{S}_{\text{Réserver un billet de train réduit}} \Rightarrow (\mathbf{S}_{\text{Présenter une carte de réduction}})$
1	0	0	1
2	0	1	1
3	1	1	1

6.3.1.4.3.2 ***L'EVALUATION DE LA CONTRAINTE DE DEPENDANCE EXCLUT***

Nous résumons dans le Tableau 14 l'évaluation de la contrainte de dépendance Exclut. En effet, la contrainte d'exclusion est satisfaite (*i.e.* $(\mathbf{S}_1 \Rightarrow \neg(\mathbf{S}_2)) = 1$) dans plusieurs cas :

- si le service $\mathbf{S}_1$ est inclus à la configuration de service ($\mathbf{S}_1$=1 à la ligne 3 du Tableau 14), alors le service $\mathbf{S}_2$ ne doit pas l'être ($\mathbf{S}_2$=0 à la ligne 3 du Tableau 14).
- si le service $\mathbf{S}_1$ ne fait pas partie de la configuration de service ($\mathbf{S}_1$=0 à la ligne 1 et ligne 2 du Tableau 14), alors le service $\mathbf{S}_2$ peut en faire partie ($\mathbf{S}_2$=1 à la ligne 2 du Tableau 14) ou non ($\mathbf{S}_2$=0 à la ligne 1 du Tableau 14).

Si les deux services $\mathbf{S}_1$ et S_2 font partie de la configuration ($\mathbf{S}_1$=S_2=1 à la ligne 4 du Tableau 14), alors la contrainte de dépendance n'est pas satisfaite (($S_1 \Rightarrow \neg(S_2)$) =0).

Autrement dit, lorsque le service $\mathbf{S}_1$ ne fait partie de la configuration, on a pas besoin d'évaluer $\mathbf{S}_2$. Néanmoins, lorsque le service $\mathbf{S}_1$ est présent dans la configuration, il est impératif d'évaluer $\mathbf{S}_2$, lequel ne doit pas faire partie de cette configuration pour que la contrainte soit satisfaite.

Tableau 14. Evaluation de la contrainte de dépendance Exclut

Ligne	S_1	S_2	$S_1 \Rightarrow \neg(S_2)$
1	0	0	**1**
2	0	1	**1**
3	1	0	**1**
4	1	1	0

EXEMPLE

Le Tableau 15 résume l'évaluation de la contrainte de dépendance $\mathbf{S}_{\text{Réserver un billet de train par internet}} \Rightarrow \neg(\mathbf{S}_{\text{Payer un billet de train par espèce}})$. Tel qu'illustré dans le Tableau 15, la présence du service $\mathbf{S}_{\text{Acheter un billet de train par internet}}$ dans la configuration empêche la présence du service $\mathbf{S}_{\text{Payer un billet de train par espèce}}$, pour que la contrainte d'exclusion soit satisfaite (ligne 3 au Tableau 15).

Aussi, cette contrainte est satisfaite lorsque le $\mathbf{S}_{\text{Réserver un billet de train par internet}}$ n'est pas présent dans la configuration (ligne 1 et ligne 2 au Tableau 15).

Tableau 15. Evaluation de $\mathbf{S}_{\text{Réserver un billet de train par internet}} \Rightarrow \neg(\mathbf{S}_{\text{Payer un billet de train par espèce}})$

Ligne	$\mathbf{S}_{\text{Réserver un billet de train par internet}}$	$\mathbf{S}_{\text{Payer un billet de train par espèce}}$	$\mathbf{S_{Réserver\ un\ billet\ de\ train\ par\ internet}} \Rightarrow \neg(\mathbf{S_{Payer\ un\ billet\ de\ train\ par\ espèce}})$
1	0	0	1
2	0	1	1
3	1	0	1

Nous avons présenté dans cette section que dans le contexte d'un service agrégat donné, des contraintes de dépendance entre ses variantes doivent être prises en compte, afin de valider l'exécutabilité de ses configurations. En effet, nous avons présenté que la présence d'un ou plusieurs services (*i.e.* les sources de la contrainte) dans une configuration peut exiger ou exclure (*i.e.* le type de la contrainte) la présence d'un autre service (*i.e.* la cible de la contrainte) de la dite configuration. Nous avons également proposé un ensemble de règles permettant de valider ces contraintes, afin que leur définition respecte d'une part la définition des contraintes, présentée plus haut (*cf.* Section 6.3.1.1), et d'autre part la sémantique des opérateurs d'agrégation. Enfin, nous avons proposé d'assimiler la logique booléenne à aux contraintes de dépendance d'exigence et d'exclusion, afin de les exploiter dans le prototype TOPSiQC (*cf.* Chapitre 9).

Comme défini dans le modèle *MiS-q*, le service intentionnel peut également fournir des qualités permettant de répondre aux exigences non-fonctionnelles des clients. Nous présentons dans la section suivante la qualité du service intentionnel.

6.3.2 LA QUALITE DU SERVICE INTENTIONNEL (QOIS)

Le quatrième aspect descriptif du service intentionnel est la qualité de service. Celle-ci est représentée par l'élément *QoiS* (*Quality of intentional Service*) à la Figure 46. Elle sert notamment à différencier les services réalisant le même but, ou encore les configurations d'un même service agrégat. L'aspect qualitatif du service intentionnel est capturé par le but qualité, ainsi que par son évaluation qualitative. La Figure 63 présente les différents concepts associés à la QoiS, à savoir la définition de la QoiS, ainsi que les différents types de QoiS. Chacun de ces concepts est présenté dans la suite de cette section.

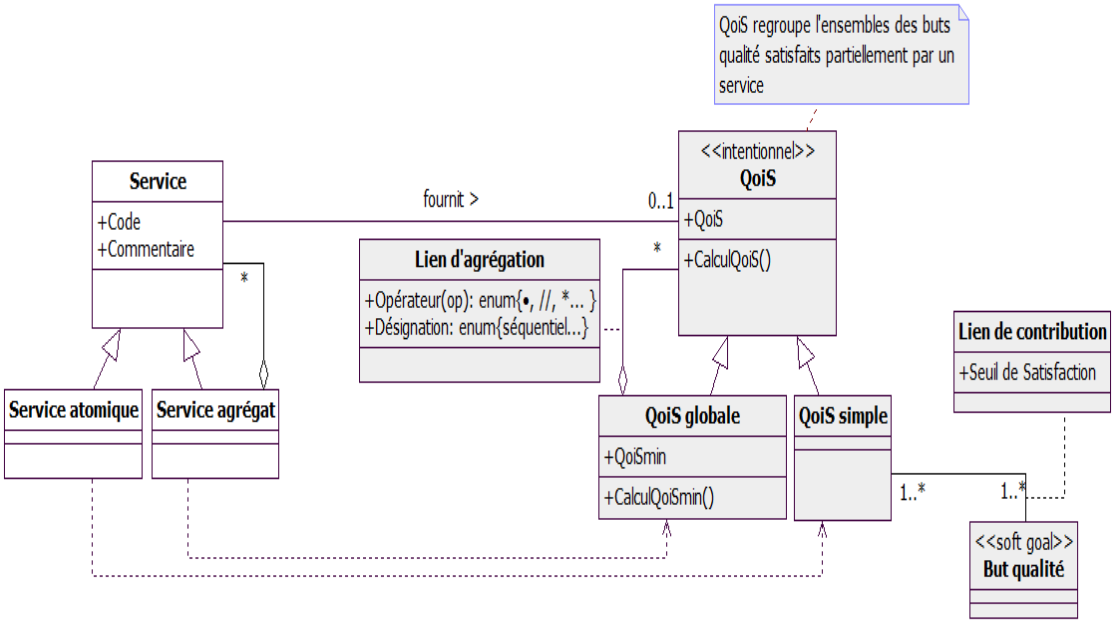

Figure 63. Le modèle de QoiS

6.3.2.1 *DEFINITION DE LA QOIS*

Nous considérons que le service intentionnel peut fournir une QoiS (cardinalité *0..1* et association entre les éléments Service et QoiS de la Figure 63). La QoiS du service intentionnel est de deux types : la *QoiS simple* et la *QoiS globale*. La QoiS simple concerne le service atomique, tandis que la QoiS globale correspond au service agrégat. Les relations entre les services (atomique et agrégat) et les QoiS (simple et globale) sont représentées par les liens de dépendance illustrées par la Figure 63. Concernant le service agrégat, une QoiS, dite $QoiS_{min}$, est également calculée. La $QoiS_{min}$ permet de publier la qualité garantie par le service agrégat, quelle que soit la configuration, et elle est également exploitée par le prototype TOPiSQC pour comparer les services agrégats des différents fournisseurs.

Pour comparer les configurations d'un même service agrégat, la $QoiS_{min}$ est inefficace, car elle privilégie les seuils négatifs de satisfaction (i.e. « - » et « -- ») aux seuils positifs (« + » et « ++ »), mettant alors sur un pied d'égalité les configurations fournissant un seul « + » et beaucoup de « - », et celles qui possèdent un seul « - » et beaucoup de « + ». Par soucis d'impartialité, nous proposons alors de calculer la qualité moyenne, nommée $QoiS_{moy}$, pour comparer les configurations du service agrégat.

Comme Rohleder (Rohleder,2009), nous définissons la qualité de service comme l'ensemble des buts qualité. Nous définissons la qualité du service intentionnel comme l'ensemble des buts qualité, auxquels le service contribue partiellement à satisfaire (*cf.* Définition 1). Cette définition est représentée à la Figure 63 par l'élément Lien de

contribution entre la QoiS simple et le but qualité. Le même but qualité peut faire partie de plusieurs QoiS. La QoiS simple est définie pour un service atomique, tandis que la QoiS globale est définie pour un service agrégat. Comme le montre la Figure 63, la QoiS globale correspond à l'agrégation des QoiS composantes (globale ou simple), et ainsi de suite jusqu'à parvenir à des QoiS simples.

Définition 1 *La QoiS est l'ensemble des buts qualité, que le service contribue partiellement à satisfaire (i.e. satisficing)*[13].

BUT QUALITE

Tel que présenté à la Figure 46, le service intentionnel réalise un but, auquel sont associés un ensemble d'états désirés (le lien entre but et état de la Figure 46 traduit cet aspect). Les états sont ceux des classes d'objets considérées dans le système. La notation pointée *NomClasse.NomAttribut = 'valeur'* est utilisée pour formuler les états. Par exemple, la réalisation du but fonctionnel *Acquérir des Produits* correspond à l'état désiré *Commande.Statut = 'créée'*, dans lequel *Commande* est le nom de la classe d'objets et *Statut* est un attribut caractéristique de commande. Le but *Acquérir des Produits* est réalisé lorsque l'état *Commande.Statut = 'créée'* est atteint. La réalisation du but d'un service intentionnel dépend donc d'un ensemble d'états désirés, correspondant aux post-conditions.

Définition 2 *Le service intentionnel réalise un but (i.e. hard goal) correspondant aux exigences fonctionnelles.*

La Définition 2 indique que le service intentionnel réalise un but, car ses critères de réalisation sont clairement définis, grâce aux post-conditions permettant de préciser et de vérifier clairement si le but du service est réalisé ou non. Par exemple, le but *Contrôler le stock* du service intentionnel $\boldsymbol{S}_{\textit{Contrôler le stock}}$ est un but, car il est facile de vérifier s'il est réalisé ou non grâce aux états des classes d'objets considérées, à savoir *Stock.état="contrôlé"* ou *Stock.état = "non contrôlé"*.

Cependant, le gestionnaire de stock peut souhaiter, par exemple, savoir si le contrôle de stock est fiable. Dans ce cas, le service intentionnel, tel que défini dans (Kaabi,2007), n'est pas capable de prendre en compte ces exigences de qualité (*i.e.* la fiabilité dans ce cas) du gestionnaire de stock. Par conséquent, nous proposons l'introduction du concept de *but qualité* dans le service intentionnel (AitAliSlimane et *al.*,2009).

Définition 3 *Le but qualité (i.e. soft goal) représente la qualité du service*

[13] La satisfaction partielle ou complète (*satisficing* en anglais) est souvent abrégée par « satisfaction » par soucis de simplicité.

intentionnel, qui correspond aux exigences non-fonctionnelles.

Le but qualité (*cf.* Chapitre 2 et Chapitre 4) correspond aux exigences non fonctionnelles, telles que la précision et la sécurité. Il correspond à la qualité que le service peut fournir, en plus de sa fonctionnalité principale (*i.e.* le but du service) (*cf.* Définition 3). Le but qualité est caractérisé par une satisfaction subjective, laquelle est liée à un contexte particulier : il convient d'un commun accord entre les clients et les fournisseurs de définir des seuils de satisfaction et accepter tout aboutissement au-delà de ces seuils. Dans l'architecture iSOA, les buts qualité et les seuils de satisfactions, associés à un domaine donné, sont définis dans le référentiel qualité

SEUIL DE SATISFACTION

Les seuils de satisfaction (*cf.* Chapitre 4) correspondent à l'évaluation qualitative de la satisfaction partielle du but qualité. Le seuil de satisfaction, présenté à la Figure 64, peut correspondre à l'une de expressions « très satisfait », « satisfait », « peu satisfait » et « pas du tout satisfait », notées respectivement par les symboles « ++ », « + », « - » et « -- » (*cf.* Définition 4).

Définition 4 *Le service contribue à « très satisfaire » (« ++ »), « à satisfaire » (« + »), à « peu satisfaire » (« - ») ou « pas du tout satisfaire » (« -- ») les buts qualité.*

Dans l'architecture iSOA, les seuils de satisfactions d'un but qualité, correspondant à un domaine particulier, sont définis d'un accord commun entre les fournisseurs et les clients, dans le référentiel qualité.

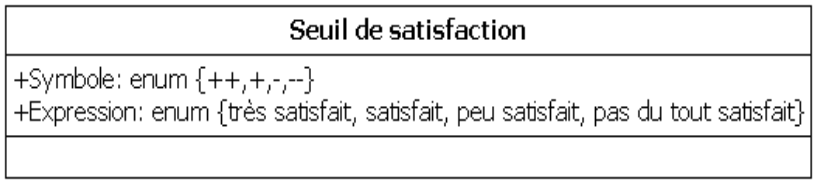

Figure 64. Les seuils de satisfaction

6.3.2.2 *TYPE DE QOIS*

La QoiS, représentant la qualité d'un service intentionnel, est dite *simple* ou *globale*. La QoiS simple correspond à la qualité fournie par le service atomique, tandis que la QoiS globale concerne le service agrégat. Voir le chapitre 4 pour plus d'information sur le service intentionnel.

La QoiS simple (*cf.* Section 6.3.2.2.1) reflète la contribution des services atomiques à la satisfaction partielle des buts qualité, tandis que la QoiS globale (*cf.* Section 6.3.2.2.2) correspond à la compilation des QoiS associées aux services composants du service agrégat. Elle est ainsi définie en fonction des opérateurs relatifs aux liens d'agrégation (*cf.* Chapitre 4). Tel que présenté à la Figure 63, la QoiS est composée de l'attribut *QoiS* correspondant à la documentation de la qualité du service intentionnel, qu'il soit atomique ou agrégat. La QoiS globale comprend l'attribut $QoiS_{min}$ qui correspond à l'ensemble minimal de buts qualité, auquel le service agrégat contribue partiellement à satisfaire. La $QoiS_{min}$ concerne uniquement le service agrégat (*cf.* Section 6.3.2.2.3). A chacune des qualités (QoiS et $QoiS_{min}$) correspond une méthode, dite *CalculQoiS()* et $CalculQoiS_{min}()$, permettant, respectivement, de calculer les QoiS et $QoiS_{min}$. Lors de la configuration du service agrégat, la qualité moyenne $QoiS_{moy}$ est calculée dynamiquement par le prototype TOPSiQC, chaque fois qu'une configuration exécutable est générée. Cette qualité est utilisée pour comparer les configurations d'un même service agrégat.

Dans la suite de ce chapitre, nous présentons ces différents types de qualité.

6.3.2.2.1 LA QOIS SIMPLE

La Figure 65 présente un extrait du modèle *MiS-q* qui met en évidence la QoiS simple associée au service atomique. La QoiS simple est calculée comme étant l'ensemble des liens de contribution existant entre le service atomique et les buts qualité.

DOCUMENTATION

Afin d'exprimer plus facilement les qualités associées à un service intentionnel, nous proposons une documentation simplifiée ainsi qu'une représentation graphique.

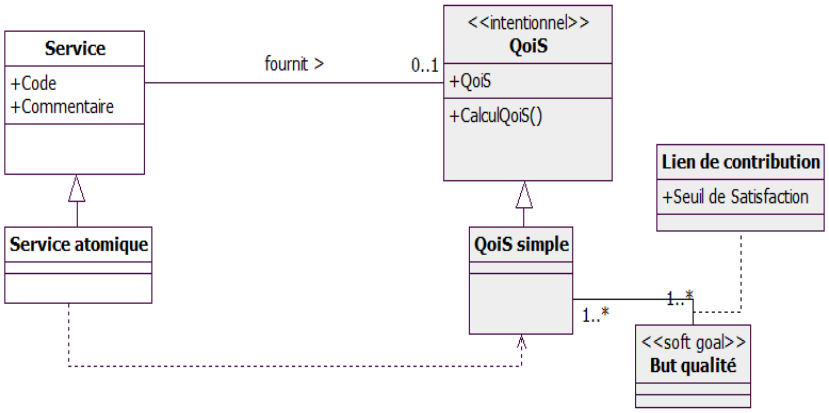

Figure 65. La QoiS simple

Soient $\boldsymbol{S}_a$ un service atomique qui contribue à la satisfaction des buts qualités q_j et les seuils de satisfaction correspondants d_j, nous adoptons la Doc 1 pour représenter la QoiS simple du service $\boldsymbol{S}_a$:

$$QoiS(\boldsymbol{S}_a) = \boldsymbol{S}_a.\{< q_j, d_j >\} \qquad \textit{Doc 1}$$

REPRESENTATION GRAPHIQUE

La représentation graphique de la QoiS simple correspond à un rectangle contenant la QoiS du service. La Figure 66 illustre la représentation graphique de la QoiS simple d'un service atomique $\boldsymbol{S}_a$.

$$\boldsymbol{QoiS}(\boldsymbol{S}_a) = \boldsymbol{S}_a.\{< qj, dj >\}$$

Figure 66. Représentation graphique de la QoiS simple

EXEMPLE

Dans l'exemple de l'approvisionnement de produits (*cf.* Chapitre 4), nous considérons que le service atomique $\boldsymbol{S}_{\textit{Contrôler le stock en continu}}$ permet de fournir la Fiabilité. Dans les exemples qui suivent, par soucis de lisibilité, nous avons choisi de ne pas associer de paramètres aux buts qualité.

La QoiS du service atomique $\boldsymbol{S}_{\textit{Contrôler le stock en continu}}$ correspond donc à l'ensemble des buts qualité auxquels ce service contribue partiellement à satisfaire. La QoiS de ce service est donc documentée comme suit :

$$QoiS(\boldsymbol{S}_{\textit{Contrôler le stock en continu}}) = \boldsymbol{S}_{\textit{Contrôler le stock en continu}}.\{< \textit{Fiabilité}, ++ >\}$$

Cette QoiS explicite que le service atomique $\boldsymbol{S}_{\textit{Contrôler le stock en continu}}$ contribue à fournir une satisfaction « ++ » du but qualité Fiabilité. Autrement dit, le contrôle de stock de façon continue permettra au gestionnaire de stock d'avoir une bonne fiabilité. La Figure 67 illustre

la représentation graphique de la QoiS du service $S_{Contrôler\ le\ stock\ en\ continu}$.

$$QoiS(S_{Contrôler\ le\ stock\ en\ continu}) = S_{Contrôler\ le\ stock\ en\ continu}.\{<Fiabilité, ++>\}$$

Figure 67. Représentation graphique de la QoiS de $S_{Contrôler\ le\ stock\ en\ continu}$

6.3.2.2.2 LA QOIS GLOBALE

La Figure 68 présente un extrait du modèle *MiS-q* qui met en évidence la QoiS globale associée au service agrégat. La QoiS globale résulte de la compilation des QoiS reliées aux services composants. Elle dépend du type de service agrégat (service composite ou service à variation) et de ses services composants. La QoiS globale est donc composée récursivement des QoiS composantes (qui peuvent être globales également, si les services composants sont agrégats), jusqu'à obtenir des QoiS simples. La QoiS simple (*cf.* Section 6.3.2.2.1) est définie comme étant l'ensemble des buts qualité que le service satisfait partiellement.

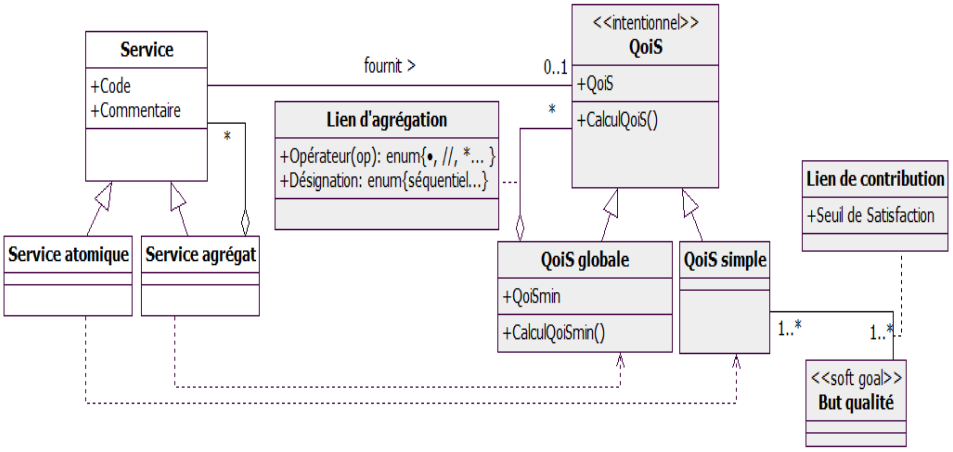

Figure 68. La QoiS globale

La composition d'un service intentionnel agrégat est dirigée par les buts (*cf.* Chapitre 4), laquelle est similaire à la décomposition de but du service. La décomposition de buts de type *Et/Ou* se traduit par l'introduction, dans le modèle *MiS-q*, de deux types de services agrégats : les services à variation qui établissent des liens *Ou* entre les services composants et offrent des choix dans la manière de réaliser le but du service ; et les services composites qui établissent des liens *Et* entre les services composants et assurent le passage d'un niveau intentionnel à un autre niveau plus proche de l'opérationnalisation. La décomposition de buts de type *Et/Ou* se traduit alors par les liens d'agrégation « • », « // », « * », « ⊗ », « v » et « ∪ », correspondant, respectivement, au « service composite séquentiel », au « service composite parallèle », au « service composite itératif », au « service à choix alternatif », au « service à choix multiple » et au « service à variation de chemin » (*cf.* Chapitre 4).

Dans le modèle *MiS-q*, l'évaluation de la qualité est elle-même orientée but. La QoiS globale du service agrégat dépend fortement des QoiS de ses services composants. Elle dépend donc des opérateurs d'agrégation « • », « // », « * », « ⊗», « v » et « ∪» (représentés par l'élément Lien agrégation à la Figure 68). La QoiS globale est définie, dans le modèle *MiS-q*, en étendant la sémantique de ces opérateurs d'agrégation. Elle peut donc concerner un service composite ou un service à variation.

La QoiS associée au service composite est dite *QoiS composite* (*cf.* Section 6.3.2.2.2.1). Elle est représentée par les opérateurs « • », « // » et « * » désignant, respectivement, la « *QoiS composite séquentielle* », « *QoiS composite parallèle* » et « *QoiS composite itérative* ». La QoiS associée au service à variation est dite *QoiS variante* (*cf.* Section 6.3.2.2.2.2). Elle est représentée par les opérateurs « ⊗», « v » et « ∪» désignant, respectivement, la « *QoiS variante à choix alternatif* », « *QoiS variante à choix multiple* » et « *QoiS variante de chemin* ».

DOCUMENTATION

De la même manière que la QoiS simple, nous proposons également une documentation et une représentation graphique de la QoiS globale.

Soient l'opérateur d'agrégation *op* (*op*∈ {•,//,⊗,v,*,∪}), le service agrégat $\boldsymbol{S}_{ag}$ correspondant à *op* ($\boldsymbol{S}_1$, $\boldsymbol{S}_2$, …, $\boldsymbol{S}_i$) et les services composants $\boldsymbol{S}_1$, $\boldsymbol{S}_2$, …, $\boldsymbol{S}_i$, nous adoptons la Doc 2 pour définir la QoiS globale d'un service agrégat :

$$QoiS\ (\boldsymbol{S}_{ag}) = QoiS\ (op\ (\boldsymbol{S}_1 \ldots \boldsymbol{S}_i)) = op\ (QoiS\ (\boldsymbol{S}_1) \ldots QoiS\ (\boldsymbol{S}_i)) \qquad \textit{Doc 2}$$

REPRESENTATION GRAPHIQUE

La représentation graphique de la QoiS globale se fait à l'aide de l'opérateur d'agrégation *op* (*op*∈ {•, //,⊗, v,*,∪}), lequel relie les différents rectangles correspondant aux QoiS des services composants.

L'évaluation et la représentation graphique de la QoiS globale se fait, tel que représentée à la Figure 69, du bas vers le haut, *i.e.* calculer d'abord les QoiS simples, ensuite remonter jusqu'aux QoiS globales, soient-elles composites (« • », « // » ou « * ») ou variantes (« ⊗ », « v » ou « ∪ »).

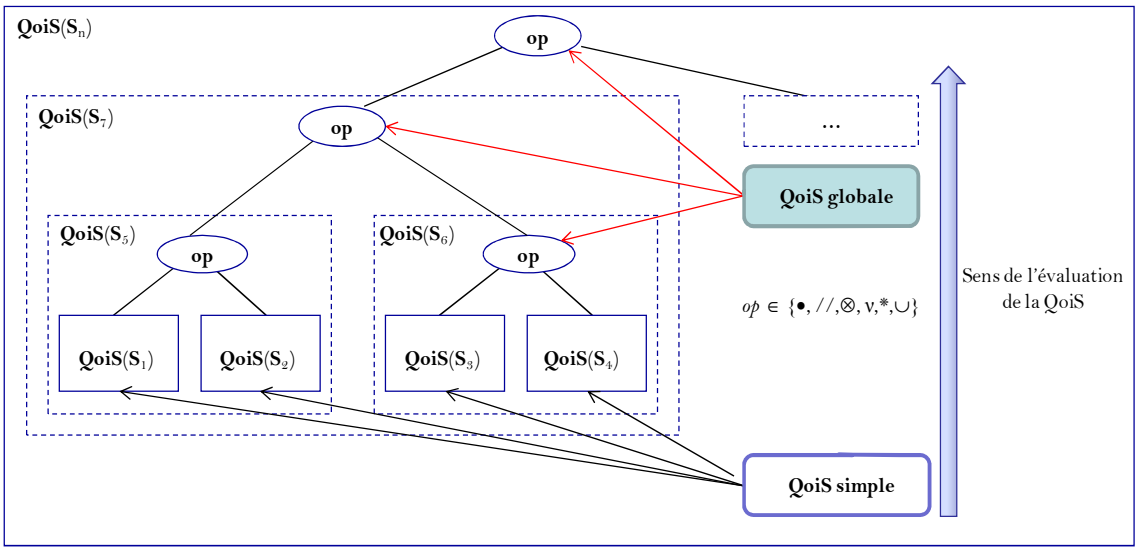

Figure 69. Représentation graphique de la QoiS globale

En effet, la Figure 69 met en évidence quatre QoiS simple, à savoir QoiS($\mathbf{S}_1$), QoiS($\mathbf{S}_2$), QoiS($\mathbf{S}_3$) et QoiS($\mathbf{S}_4$). Ces dernières sont compilées en QoiS($\mathbf{S}_5$) et QoiS($\mathbf{S}_6$) selon "op" ($op \in \{\bullet, //, \otimes, \vee, *, \cup\}$), l'opérateur d'agrégation en question. Les QoiS($\mathbf{S}_5$) et QoiS($\mathbf{S}_6$) sont également compilées en QoiS($\mathbf{S}_7$), ainsi de suite jusqu'à arriver à la QoiS racine, à savoir QoiS($\mathbf{S}_n$). La documentation de la QoiS simple se fait selon la Doc 1, tandis que celle de la QoiS globale (*i.e.* QoiS($\mathbf{S}_5$), QoiS($\mathbf{S}_6$), QoiS($\mathbf{S}_7$) et QoiS($\mathbf{S}_n$)) se fait suivant la Doc 2.

La QoiS composite et la QoiS variante se basent donc sur la sémantique des opérateurs et utilisent la Doc 2 pour, respectivement, évaluer et représenter la QoiS globale du service agrégat. Dans ce qui suit, une présentation plus détaillée de ces deux QoiS est proposée, à savoir la QoiS composite (*cf.* Section 6.3.2.2.2.1) et la QoiS variante (*cf.* Section 6.3.2.2.2.2).

6.3.2.2.2.1 *LA QOIS COMPOSITE*

La QoiS composite s'appuie sur la décomposition de type *Et* du service composite (*cf.* Chapitre 4). Nous considérons, comme le montre la Figure 70, qu'il existe un parallélisme entre la décomposition du service composite $\mathbf{S}_{B1}$ et l'évaluation de ce même service, à savoir la QoiS($\mathbf{S}_{B1}$). Rappelons que si le service $\mathbf{S}_{B1}$ se décompose par le lien *Et*, en services composants $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$, il est accepté que le service $\mathbf{S}_{B1}$ est considéré fourni que si les services composants $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$ le sont (Kaabi,2007). De même, la QoiS($\mathbf{S}_{B1}$) ne peut être évaluée ou calculée que si la qualité des services composants $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$ (*i.e.* QoiS($\mathbf{S}_{B1.1}$), QoiS($\mathbf{S}_{B1.2}$) et QoiS($\mathbf{S}_{B1.3}$)) le sont aussi.

La QoiS composite est évaluée à partir de « toutes » les QoiS des services composants. Le parallélisme entre les services et leurs QoiS nous conduit à étendre la sémantique des opérateurs de composition « • », « // » ou « * », qui traduisent, au niveau des QoiS, les

relations d'assemblage de la QoiS composite (fournie par le service composite) à partir des QoiS composantes (fournies par les services composants).

Dans ce qui suit, nous définissons et nous illustrons par un exemple les différentes QoiS composites, à savoir la QoiS composite séquentielle (« • »), la QoiS composite parallèle (« // ») et la QoiS composite itérative (« * »).

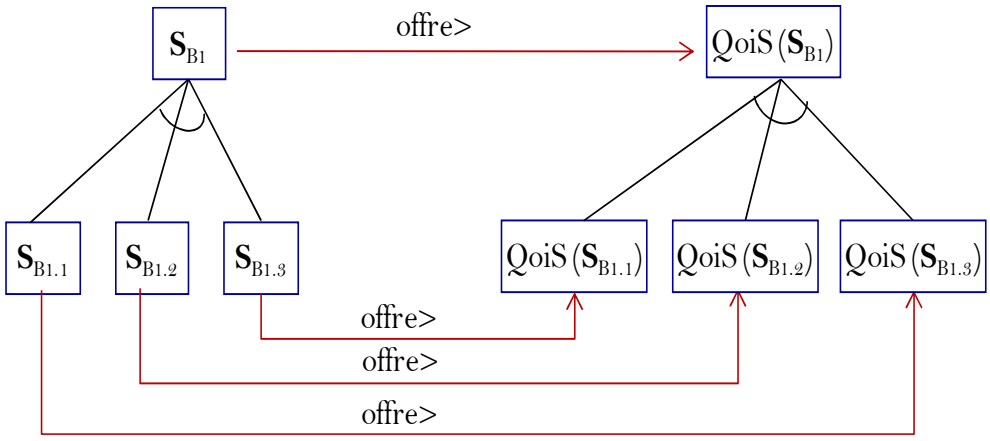

Figure 70. Parallélisme service composite/QoiS composite

LA QOIS COMPOSITE SEQUENTIELLE

La QoiS associée au service composite séquentiel (*cf.* Chapitre 4), notée « • », est dite *QoiS composite séquentielle*. Le service composite séquentiel nécessite un ordre d'exécution précis entre ses services composants. Sa QoiS correspond donc à la qualité que fournit, séquentiellement, chacun des services composants exécutés. Soient le service composite séquentiel $\mathbf{S}_{B1}$ et les services composants $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$, la QoiS de $\mathbf{S}_{B1}$ correspond à la QoiS de $\mathbf{S}_{B1.1}$, à celle de $\mathbf{S}_{B1.2}$ et à la QoiS de $\mathbf{S}_{B1.3}$. En effet, les services $\mathbf{S}_{B1.1,}$ $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$ sont fournis successivement, ainsi que leurs QoiS respectives.

EXEMPLE

Dans l'exemple de l'approvisionnement de produits, nous considérons que l'introduction des produits livrés en stock correspond au service séquentiel $\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}$=•($\mathbf{S}_{Acquérir\ des\ produits}$, $\mathbf{S}_{Entrer\ en\ stock\ les\ produits\ livrés}$). Ce dernier nécessite la réalisation du but *Entrer en stock les produits livrés*, lequel ne peut se faire que si la commande des produits est préalablement effectuée, via le but *Acquérir des produits*. Dans ce cas, la QoiS composite séquentielle du service $\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}$ correspond, d'abord, à la QoiS fournie par le service $\mathbf{S}_{Acquérir\ des\ produits}$ (*i.e.* la QoiS fournie par le premier service exécuté), ensuite à celle proposée par le service $\mathbf{S}_{Entrer\ en\ stock\ les\ produits\ livrés}$ (*i.e.* la QoiS fournie par le second service exécuté). Ceci se traduit par la QoiS composite séquentielle suivante :

$$QoiS\ (\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}) = \bullet\ (QoiS\ (\mathbf{S}_{Acquérir\ des\ produits}),\ QoiS\ (\mathbf{S}_{Entrer\ en\ stock\ les\ produits\ livrés}))$$

La Figure 71 montre la représentation graphique de la QoiS composite séquentielle du service $\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}$, à savoir la $QoiS\ (\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement})$.

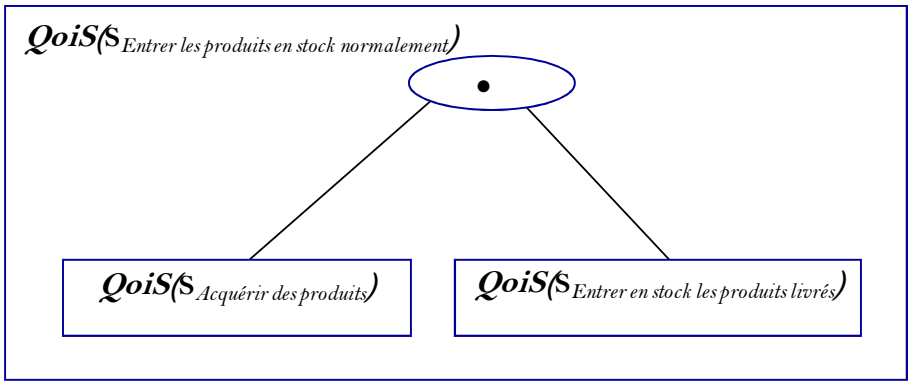

Figure 71. Représentation graphique de la QoiS du service $\boldsymbol{S}_{Entrer\ les\ produits\ en\ stock\ normalement}$

LA QOIS COMPOSITE PARALLELE

La *QoiS composite parallèle*, notée « // », est, comme son nom l'indique, reliée au service composite parallèle (*cf.* Chapitre 4). Ce dernier peut être exécuté selon un ordre quelconque et non dans un ordre impératif. Sa QoiS correspond ainsi à la qualité que fournit, parallèlement, chacun des services composants exécutés. Soient le service composite parallèle $\mathbf{S}_{B1}$ et les services composants $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$. Ces derniers sont fournis de façon parallèle, *i.e.* les services $\mathbf{S}_{B1.1,}$ $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$ sont fournis simultanément (au lieu de séquentiellement), ainsi que leurs QoiS respectives.

EXEMPLE

Nous pouvons considérer que contrôler le stock par valuation et par inspection qualité peut correspondre au service composite parallèle $\boldsymbol{S}_{Contrôler\ le\ stock} = //\ (\boldsymbol{S}_{Valuer\ le\ stock}, \boldsymbol{S}_{Inspecter\ la\ qualité\ du\ stock})$, car les deux contrôles peuvent être exécutées en parallèle. Pour cet exemple, la QoiS composite parallèle du service $\boldsymbol{S}_{Contrôler\ le\ stock}$ correspond alors aux QoiS fournies, simultanément (au lieu de séquentiellement), par les deux services $\boldsymbol{S}_{Valuer\ le\ stock}$ et $\boldsymbol{S}_{Inspecter\ la\ qualité\ du\ stock}$. Ceci se traduit par la QoiS composite parallèle suivante :

$$QoiS\ (\boldsymbol{S}_{Contrôler\ le\ stock}) = //\ (QoiS\ (\boldsymbol{S}_{Valuer\ le\ stock}),\ QoiS\ (\boldsymbol{S}_{Inspecter\ la\ qualité\ du\ stock}))$$

La Figure 72 montre la représentation graphique de la QoiS composite parallèle du service $\boldsymbol{S}_{Contrôler\ le\ stock}$.

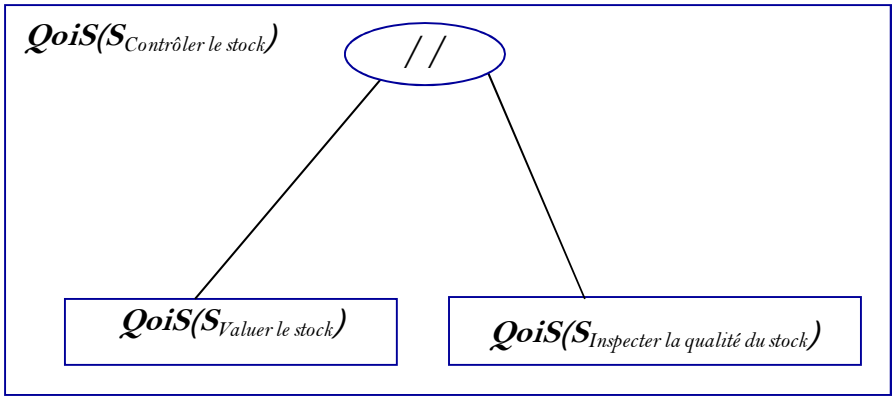

Figure 72. Représentation graphique de la QoiS du service $S_{Contrôler\ le\ stock}$

LA QOIS COMPOSITE ITERATIVE

La QoiS associée au service composite itératif (*cf.* Chapitre 4), notée « * », est dite *QoiS composite itérative*. L'exécution itérative d'un service peut s'appliquer aussi bien à un service atomique qu'à un service agrégat. Sa QoiS correspond, répétitivement, à la qualité fournie par les services exécutés de façon itérative. Soit le service composite itératif $\mathbf{S}^*_{B1}$, celui-ci est fourni de façon répétitive; *i.e.* il peut être exécuté de façon itérative jusqu'à ce que le résultat escompté soit atteint. Ainsi, la QoiS de $\mathbf{S}^*_{B1}$ correspond, itérativement, à la QoiS de $\mathbf{S}_{B1}$.

EXEMPLE

Dans l'exemple de l'approvisionnement de produits, le service permettant d'effectuer l'approvisionnement de produits est un service composite nommé $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}=\bullet(\mathbf{S}_{Recevoir\ le\ stock}, \mathbf{S}^*_{Contrôler\ le\ stock}, \mathbf{S}_{Contrôler\ le\ paiement\ des\ factures})$. Le service composant itératif $\mathbf{S}^*_{Contrôler\ le\ stock}$ est exécuté de manière répétitive de façon à contrôler le stock chaque fois que c'est nécessaire. La QoiS du service $\mathbf{S}^*_{Contrôler\ le\ stock}$ correspond alors aux QoiS fournies, itérativement, par le service lui même. La QoiS composite itérative du service $\mathbf{S}^*_{Contrôler\ le\ stock}$ est définie comme suit :

$$QoiS\ (\mathbf{S}^*_{Contrôler\ le\ stock}) = (QoiS\ (\mathbf{S}_{Contrôler\ le\ stock}))^*$$

D'où la QoiS composite séquentielle du service $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$, qui est définie comme suit:

$$QoiS\ (\mathbf{S}_{\text{Effectuer l'approvisionnement en produits}}) = \bullet\ (QoiS\ (\mathbf{S}_{\text{Recevoir le stock}}), QoiS\ (\mathbf{S}^*_{Contrôler\ le\ stock}),$$
$$QoiS\ (\mathbf{S}_{\text{Contrôler le paiement des factures}}))$$

La Figure 73 montre la représentation graphique de la QoiS composite itérative.

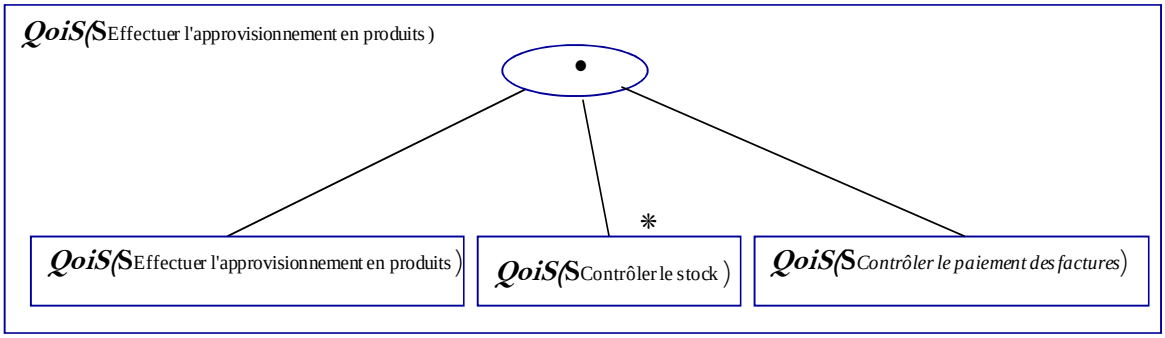

Figure 73. Représentation graphique de la QoiS du service **S**Effectuer l'approvisionnement en *produits*

6.3.2.2.2.2 ***LA QOIS VARIANTE***

Une QoiS variante reflète la variabilité dans la modélisation de la qualité d'un service intentionnel. Cette variabilité est justifiée par le besoin de flexibilité dans l'exécution du service intentionnel. En effet, le service à variation permet de réaliser le même but, mais ses variantes (*i.e.* les chemins du service agrégat) peuvent satisfaire différemment (*i.e.* avec des seuils de satisfaction différents) divers buts qualité.

La QoiS variante s'appuie sur la décomposition de type *Ou* du service à variation. Nous considérons, comme le montre la Figure 74, qu'il existe un parallélisme entre la décomposition du service à variation $\mathbf{S}_{B1}$ et l'évaluation de ce même service, à savoir la QoiS($\mathbf{S}_{B1}$). Rappelons que si le service $\mathbf{S}_{B1}$ se décompose par le lien *Ou*, en services composants $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$, il est accepté que le service $\mathbf{S}_{B1}$ est considéré fourni que si l'un de ses services composants $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ et $\mathbf{S}_{B1.3}$ l'est (Kaabi,2007). De même, la QoiS($\mathbf{S}_{B1}$) ne peut être évaluée ou calculée que si l'une des qualités QoiS($\mathbf{S}_{B1.1}$), QoiS($\mathbf{S}_{B1.2}$) et QoiS($\mathbf{S}_{B1.3}$) l'est également.

Ainsi, la QoiS variante peut dépendre aussi bien « d'une » (lorsqu'il s'agit par exemple d'un service à choix alternatif) que de "toutes" (lorsqu'il s'agit d'un service à choix multiple) les QoiS des services composants. Le parallélisme effectué entre le service à variation et ses QoiS nous conduit à étendre la sémantique des opérateurs de variation « $\otimes$ », « v » ou « $\cup$ », qui traduisent, au niveau des QoiS, les relations d'assemblage de la QoiS variante (fournie par le service à variation) à partir des QoiS composantes (fournies par les services composants).

La qualité variante permet de différencier les différents services $\mathbf{S}_{B1.1}$, $\mathbf{S}_{B1.2}$ ou $\mathbf{S}_{B1.3}$, lesquels réalisent le même but (*i.e.* B_1) et pouvant offrir des qualités différentes QoiS($\mathbf{S}_{B1.1}$), QoiS($\mathbf{S}_{B1.2}$) ou QoiS($\mathbf{S}_{B1.3}$). Par conséquent, nous proposons que la sélection du service intentionnel dépende de la qualité qu'il fournit.

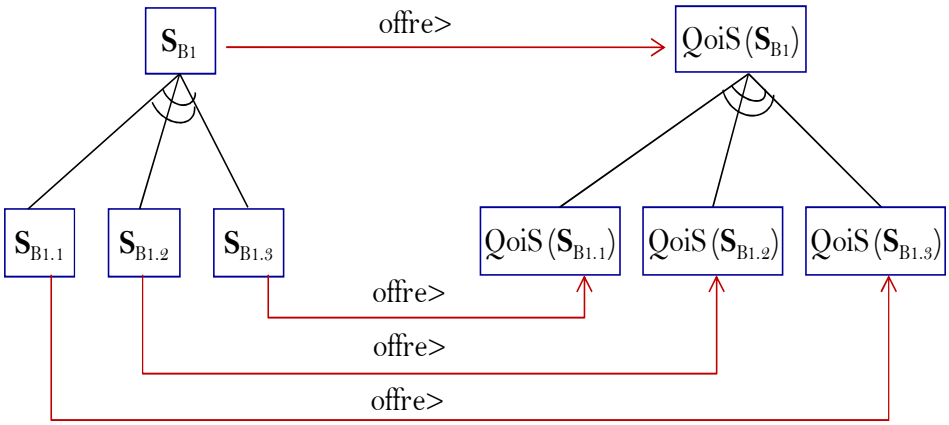

Figure 74. Parallélisme service à variation/QoiS variante

Dans ce qui suit, nous définissons et nous illustrons par un exemple les différentes QoiS variantes, à savoir la QoiS variante à choix alternatif (« ⊗ »), la QoiS variante à choix multiple (« *v* ») et la QoiS variante de chemin (« ∪ »).

LA QOIS VARIANTE A CHOIX ALTERNATIF

La *QoiS variante à choix alternatif*, notée « ⊗ », est associée, comme son nom l'indique, au service à choix alternatif (*cf.* Chapitre 4). Elle correspond à une relation exclusive (*Ou exclusif*) entre les QoiS des services composants alternatifs, *i.e.* une et une seule QoiS offerte par l'un des services composants alternatifs est retenue (correspondant en réalité à la QoiS offerte par le service retenu).

EXEMPLE

Toujours dans l'exemple de l'approvisionnement de produits, nous pouvons considérer que l'acquisition des produits par planification correspond en réalité au service à variation **S**Acquérir des produits par planification=⊗ (**S**Commander des produits par seuil de réapprovisionnement, **S**Commander des produits par prévisions stratégiques). La QoiS du service **S**Acquérir des produits par planification correspond, dans ce cas, à la QoiS qui sera fournie exclusivement par l'un de ses services composants, à savoir la QoiS de **S**Commander des produits par seuil de réapprovisionnement ou (exclusif) la QoiS de **S**Commander des produits par prévisions stratégiques. Ceci se traduit par la QoiS variante à choix alternatif suivante :

*QoiS (***S**Acquérir des produits par planification*)* = ⊗ *(QoiS (***S**Commander des produits par seuil de réapprovisionnement*),*

*QoiS (***S**Commander des produits par prévisions stratégiques*))*

Soient, les services **S**Commander des produits par seuil de réapprovisionnement et **S**Commander des produits par prévisions stratégiques, lesquels contribuent, respectivement, « + » et « ++ » à la satisfaction du *Performance*. La Figure 75, avec la documentation des buts qualité, montre la représentation graphique de la QoiS du service **S**Acquérir des produits par planification.

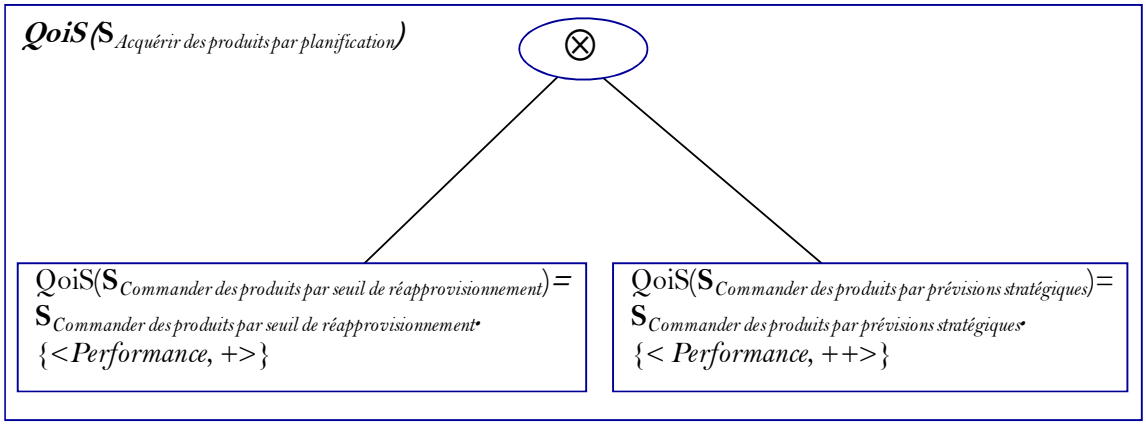

Figure 75. Représentation graphique du service $\mathbf{S}_{\text{Acquérir des produits par planification}}$

LA QOIS VARIANTE A CHOIX MULTIPLE

La QoiS associée au service à choix multiple (*cf.* Chapitre 4), notée « *v* », est dite *QoiS variante à choix multiple*. Elle correspond à une relation non exclusive (*Ou*) entre les QoiS des services composants alternatifs, *i.e.* une ou plusieurs QoiS (correspondant aux services composants) peut être choisie parmi celles offertes par les services composants alternatifs. En fait, le choix d'un ou plusieurs services composants, réalisant le même but fonctionnel, dépend de la QoiS fournie par ces services.

EXEMPLE

Dans l'exemple d'approvisionnement de produits, nous considérons que l'acquisition de produits peut se faire de deux manières non exclusives. Ceci correspond au service variant à choix multiple $\mathbf{S}_{\text{Acquérir des produits}} = \nu\ (\mathbf{S}_{\text{Commander des produits manuellement}}, \mathbf{S}_{\text{Acquérir des produits par planification}})$.

La QoiS du service $\mathbf{S}_{\text{Acquérir des produits}}$ correspond soit à la QoiS de l'un de ses services composants (si l'un d'eux est choisi), soit à celle de plusieurs de ses services composants (si plusieurs sont choisis). Dans ce dernier cas, la QoiS variante à choix multiple se comporte comme une QoiS composite. Ceci se traduit par la QoiS variante à choix multiple suivante :

$$QoiS\ (\mathbf{S}_{\text{Acquérir des produits}}) = \nu\ (QoiS\ (\mathbf{S}_{\text{Commander des produits manuellement}}), QoiS\ (\mathbf{S}_{\text{Acquérir des produits par planification}}))$$

La Figure 76 montre la représentation graphique de la QoiS du service $\mathbf{S}_{\text{Acquérir des produits}}$.

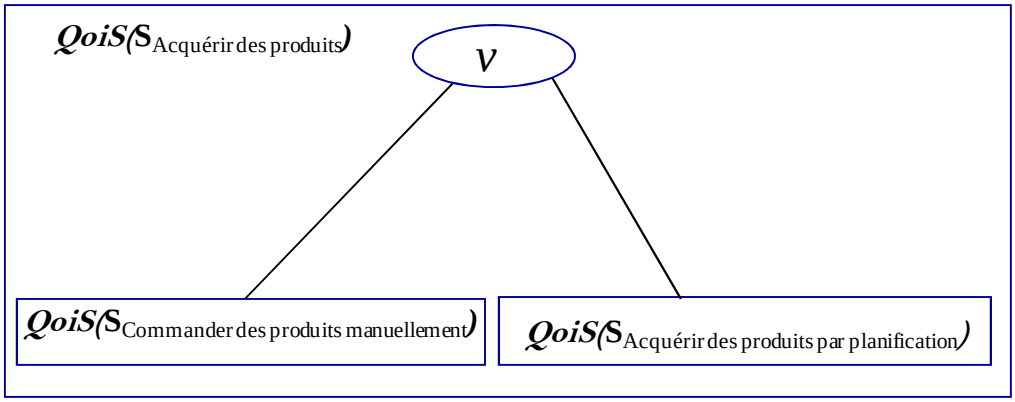

Figure 76. Représentation graphique de la QoiS du service $\mathbf{S}_{\text{Acquérir des produits}}$

LA QOIS VARIANTE DE CHEMIN

La QoiS variante de chemin, notée « ∪», est définie lorsque nous sommes en présence d'un service à variation de chemin (*cf.* Chapitre 4). Elle correspond à une relation exclusive (*Ou exclusif*) entre les QoiS des chemins alternatifs de buts, *i.e.* une et une seule QoiS (correspondant à la QoiS associée aux services dans un chemin précis) peut être choisie parmi celles offertes par les différents chemins alternatifs.

EXEMPLE

Dans l'exemple de l'approvisionnement de produits, nous définissons le service $\mathbf{S}_{Recevoir\ le\ stock} = \cup(\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ directement}, \mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement})$ comme un service à variation de chemin. La QoiS du service $\mathbf{S}_{Recevoir\ le\ stock}$ consiste à choisir entre la QoiS du chemin $\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ directement}$ et la QoiS du chemin $\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}$. Ceci est traduit par la QoiS variante de chemin suivante :

$$QoiS\ (\mathbf{S}_{Recevoir\ le\ stock}) = \cup\ (QoiS\ (\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ directement}),\ QoiS\ (\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}))$$

La Figure 77 montre la représentation graphique de la QoiS du service $\mathbf{S}_{Recevoir\ le\ stock}$. En effet, la QoiS du chemin 1 est représentée par la $QoiS(\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement})$ et la QoiS du chemin 2 correspond à la $QoiS(\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ directement})$. La QoiS du chemin 1 est elle-même une QoiS composite séquentielle, à savoir • (QoiS($\mathbf{S}_{\text{Acquérir des produits}}$), QoiS($\mathbf{S}_{Entrer\ en\ stock\ les\ produits\ livrés}$)). Selon la Figure 77, la QoiS($\mathbf{S}_{\text{Acquérir des produits}}$) est une QoiS variante à choix multiple correspondant à $QoiS\ (\mathbf{S}_{\text{Acquérir des produits}}) = v\ (QoiS\ (\mathbf{S}_{\text{Commander des produits manuellement}}), QoiS\ (\mathbf{S}_{\text{Acquérir des produits par planification}}))$.

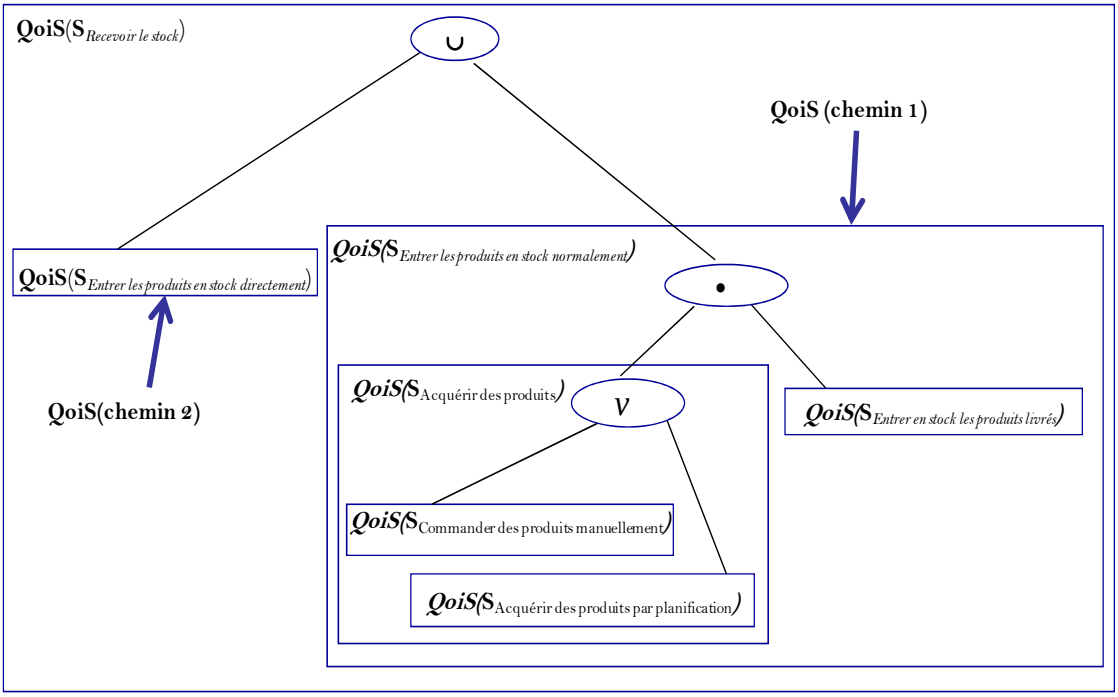

Figure 77. Représentation graphique de la QoiS du service $\mathbf{S}_{Recevoir\ le\ stock}$

Nous avons définie plus haut (*cf.* Section 6) que la QoiS d'un service intentionnel peut être simple ou globale : la QoiS simple est reliée au service atomique, tandis que la QoiS globale est associée au service agrégat. La QoiS globale définie par la Doc 2 (*cf.* Section 6.3.2.2.2) est certes informative, mais reste d'une part difficile à être interprétée par les agents métiers. D'autre part, elle ne peut pas être exploitée directement par le processus de sélection de services, que nous proposons dans cette thèse (*cf.* Chapitre 8). Pour remédier à ce problème, nous proposons de calculer la qualité minimale, dite la $QoiS_{min}$, que peut fournir un service agrégat. Nous présentons, dans la prochaine section, la définition et le calcul de la $QoiS_{min}$.

6.3.2.2.3 LA QUALITE MINIMALE : $QOIS_{MIN}$

La qualité globale dépend des différents opérateurs d'agrégation « • », « // », « * », « ⊗ », « *v* » et « ∪ », définis précédemment (*cf.* Section 6.3.2.2.2). La QoiS globale est donc composite ou variante, et elle peut varier (augmenter ou diminuer) selon les variantes du service choisie. La QoiS globale, telle que définie par la Doc 2 (*cf.* Section 6.3.2.2.2), permet de documenter la qualité du service agrégat, en précisant les services composants satisfaisant (*i.e.* seuil de satisfaction égale à « + » ou « ++ ») des qualités, ainsi que ceux qui ne les satisfaisant pas (*i.e.* seuil de satisfaction égale à « - » ou « -- »).

Par exemple, la QoiS du service $\boldsymbol{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ est définie comme suit : $QoiS\ (\boldsymbol{S}_{Effectuer\ l'approvisionnement\ en\ produits}) = \bullet\ (v\ (\otimes\ (QoiS(\mathbf{S}_{\text{Commander des produits par seuil de}}$

réapprovisionnement*)*, *QoiS(***S**Commander des produits par prévisions stratégiques*)),QoiS(***S**Commander des produits manuellement*)),* *v(QoiS(**S***Inspecter la qualité du stock*),* *⊗(QoiS(**S***Etablir un inventaire par échantillonnage*),* *QoiS(**S***Etablir un inventaire périodiquement*), QoiS(**S***Etablir un inventaire en continu*))), QoiS(**S**Contrôler le paiement des factures*))*. La QoiS du service $\boldsymbol{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ est égale aussi à : *QoiS (**S**Effectuer l'approvisionnement en produits*) = ● *(v (⊗ (**S**Commander des produits par seuil de réapprovisionnement. <Performance,+>, **S**Commander des produits par prévisions stratégiques. <Performance,++>), **S**Commander des produits manuellement. <Performance,->)), v (**S***Inspecter la qualité du stock. <Fiabilité,+>, ⊗(**S***Etablir un inventaire par échantillonnage. <Fiabilité,->, **S***Etablir un inventaire périodiquement. <Fiabilité,+>, **S***Etablir un inventaire en continu. <Fiabilité,+++>))), **S**Contrôler le paiement des factures. <Fiabilité,+++>).

Néanmoins, la QoiS globale définie par la Doc 2 (*cf.* Section 6.3.2.2.2) ne peut pas être exploitée directement par l'algorithme de sélection de services que nous proposons. Pour compléter la QoiS globale définie par la Doc 2 (*cf.* Section 6.3.2.2.2), nous proposons de calculer la qualité minimale, dite la $QoiS_{min}$. Celle-ci indique la qualité garantie par le service agrégat, sachant que ce dernier pourrait fournir une meilleure qualité en fonction de ses configurations (*cf.* Chapitre 8). La $QoiS_{min}$ est exploitée par le processus de sélection de services, afin de choisir parmi plusieurs services (*cf.* Chapitre 8).

6.3.2.2.3.1 *DEFINITION DE LA $QOIS_{MIN}$*

Nous considérons l'exemple de l'approvisionnement de produits, dans lequel la QoiS du service $\mathbf{S}_{Recevoir\ le\ stock}$ est définie selon la Doc 2, comme suit :

$$QoiS\ (\mathbf{S}_{Recevoir\ le\ stock}) = \cup\ (QoiS\ (\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ directement}),\ QoiS\ (\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}))$$

La QoiS($\mathbf{S}_{Recevoir\ le\ stock}$) explicite que la qualité du service $\mathbf{S}_{Recevoir\ le\ stock}$ est une relation exclusive entre la qualité des deux chemins : $\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ directement}$ et $\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement}$. Néanmoins, cette QoiS ne précise pas les buts qualité et les seuils de satisfaction que ce service permet de fournir, d'où le besoin de la $QoiS_{min}$. Autrement dit, la $QoiS_{min}(\mathbf{S}_{Recevoir\ le\ stock})$ devrait indiquer l'ensemble des buts qualité, auxquels les services contribuent partiellement à satisfaire. La $QoiS_{min}(\mathbf{S}_{Recevoir\ le\ stock})$ est calculée à partir des QoiS composantes, à savoir la QoiS $(\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ directement})$ et la QoiS$(\mathbf{S}_{Entrer\ les\ produits\ en\ stock\ normalement})$. Si ces dernières sont des QoiS globales, alors il faut également calculer leurs $QoiS_{min}$, ainsi de suite jusqu'à arriver à des QoiS simples.

DOCUMENTATION

De la même manière que la QoiS (simple et globale), nous proposons une documentation et

une représentation graphique de la $QoiS_{min}$.

Soient $\boldsymbol{S}_i$ le service intentionnel contribuant à la satisfaction partielle des buts qualités q_j et les seuils de satisfaction correspondants d_j, nous adoptons la Doc 3 pour représenter la $QoiS_{min}$ du service $\boldsymbol{S}_i$:

$$QoiS_{min}\ (\boldsymbol{S}_i) = \{<Q, D, \{\sigma\}>\} \qquad \textit{Doc 3}$$

Q correspond à l'ensemble minimal des buts qualités (q_j) que le service $\boldsymbol{S}_i$ contribue partiellement à satisfaire ; *D* correspond aux seuils minimaux de satisfaction correspondant à chaque but qualité ; et {σ} définit les services composants offrant *Q* et *D*.

REPRESENTATION GRAPHIQUE

La Figure 78 résume la représentation graphique de la $QoiS_{min}$. La $QoiS_{min}$ peut concerner une QoiS composite (« • », « // » ou « * ») ou une QoiS variante (« ⊗ », « *v* », « ∪ »), selon l'opérateur d'agrégation « *op* ». La QoiS simple et la $QoiS_{min}$ sont notées, respectivement, à l'aide de la *Doc 1* (*cf.* Section 6.3.2.2.1) et de la *Doc 3* (*cf.* Section 6.3.2.2.3).

Le calcul de la $QoiS_{min}$ s'apparente à la propagation de la satisfaction minimale des buts qualité. La Figure 78 illustre un exemple de calcul de la $QoiS_{min}$. Celui-ci se fait du bas (*i.e.* des QoiS simples) vers le haut (*i.e.* des QoiS globales), en tenant en compte de la QoiS des services atomiques et remonter ensuite cette information vers les services agrégats.

En effet, la Figure 78 met en évidence six QoiS simples, à savoir $QoiS(\mathbf{S}_1)$, $QoiS(\mathbf{S}_2)$, $QoiS(\mathbf{S}_3)$, et $QoiS(\mathbf{S}_4)$. Ces dernières sont agrégées, en fonction de l'opérateur « *op* » ($op \in \{\bullet, //, \otimes, v, *, \cup\}$), en $QoiS_{min}(\mathbf{S}_5)$ et $QoiS_{min}(\mathbf{S}_6)$. Ces dernières sont également compilées en $QoiS_{min}(\mathbf{S}_7)$ et ainsi de suite jusqu'à arriver à la $QoiS_{min}$ racine, à savoir $QoiS_{min}$ ($\mathbf{S}_n$).

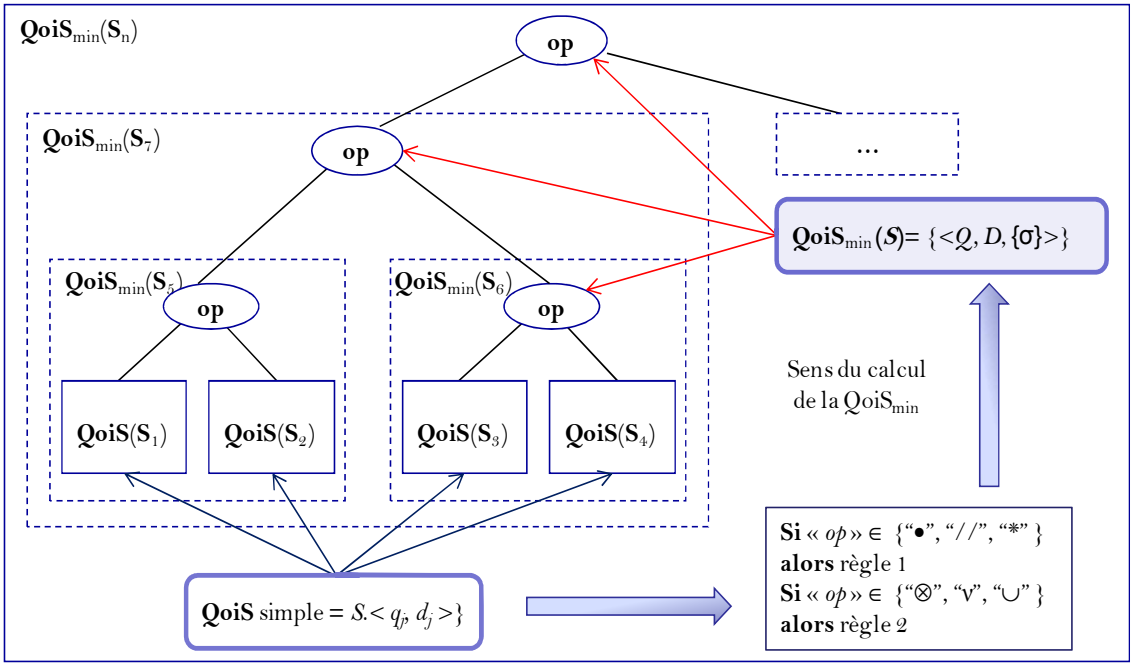

Figure 78. Représentation graphique de la QoiS$_{min}$ et de son calcul

Une fois les QoiS simples définies, le calcul de la QoiS$_{min}$ se fait en appliquant la règle 1 (*cf.* Section 6.3.2.2.3.2) dans le cas d'un service composite (« • », « // » ou « * ») et la règle 2 (*cf.* Section 6.3.2.2.3.3) dans le cas d'un service à variation (« ⊗ », « ν », « ∪ »). Ces deux règles (*i.e.* règle 1 et règle 2) sont décrites, dans plus de détail, dans ce qui suit.

6.3.2.2.3.2 *LA QOIS$_{MIN}$ COMPOSITE*

Dans le cas d'un service composite (*i.e.* « • », « // », « * »), nous adoptons la Doc 3 pour représenter la QoiS$_{min}$ du service composite S_{comp}. La Doc 3 exprime la qualité fournie par le service composite, en termes de buts qualité que le service S_{comp} contribue à satisfaire partiellement (*i.e.* Q), ainsi que les seuils de satisfaction (*i.e.* D) correspondant. Elle définit également les services composants fournissant cette qualité minimale (*i.e.* $\{\sigma\}$). Telle que le montre la Doc 3, la QoiS$_{min}$ composite est représentée par trois éléments.

Le premier élément se base sur les principes de la théorie des ensembles. En fait, telle que définie plus haut, la QoiS composite reflète un lien d'évaluation de type « *Et* » entre les QoiS composantes (*cf.* Section 6.3.2.2.2.1). Soient $\boldsymbol{S_1}$ et $\boldsymbol{S_2}$ deux services composants du service composite $\boldsymbol{S_{comp}}$. Soient $A= \{q_1, q_2\}$ et $B= \{q_1, q_3\}$ les ensembles de buts qualité satisfaits partiellement par $\boldsymbol{S_1}$ et $\boldsymbol{S_2}$. L'exécution de $\boldsymbol{S_{comp}}$ implique l'exécution de tous les services composants, à savoir $\boldsymbol{S_1}$ et $\boldsymbol{S_2}$. L'exécution de $\boldsymbol{S_1}$ fournit la QoiS correspondant à A et l'exécution de $\boldsymbol{S_2}$ procure la QoiS correspondant à B. Ceci implique que le service agrégat composite $\boldsymbol{S_{comp}}$ va bénéficier des qualités des services composants $\boldsymbol{S_1}$ « *Et* » $\boldsymbol{S_2}$. Par

conséquent, la $QoiS_{min}$ de $\boldsymbol{S}_{comp}$ correspond à l'union des deux ensembles *A* et *B* (*i.e. A U B*). Ainsi, la $QoiS_{min}(\mathbf{S}_{comp}) = \{q_1, q_2, q_3\}$. Ceci se traduit par :

$$\forall q_j, q_j \in A \cup B \equiv \left(\left(q_j \in A\right) \vee \left(q_j \in B\right)\right)$$

Le premier élément de la Doc 3 correspond donc à la réunion des buts qualité fournis par les services composants. Elle est calculée comme suit :

$$Q = \bigcup_{j=1}^{m} \left\{q_j\right\}$$

Le deuxième élément de la Doc 3 consiste à définir, pour chaque but qualité, le seuil minimal de satisfaction. Soient S_1 et S_2 les services composants du service composite S_{comp}. Soient $A = \{q_1, q_2\}$ et $B = \{q_1, q_3\}$ les ensembles de buts qualité satisfaits partiellement par S_1 et S_2. La $QoiS_{min}$ de S_{comp} correspond à l'union des deux ensembles *A* et *B*, *i.e.* la $QoiS_{min}(S_{comp}) = \{q_1, q_2, q_3\}$. Cependant, S_1 fournit q_1 *et* q_2 avec les seuils de satisfaction respectifs d_1 et d_2. De façon identique, S_2 fournit q_1 *et* q_3 avec les seuils de satisfaction respectifs d_3 et d_4. Aussi, $A = \{<q_1, d_1>, <q_2, d_2>\}$ et $B = \{<q_1, d_3>, <q_3, d_4>\}$. Les seuils de satisfaction d_1, d_2, d_3 et d_4 correspondent aux valeurs « ++ », « + », « - » ou « -- ». Par conséquent, il s'agit de définir les seuils de satisfaction de $QoiS_{min}$ (S_{comp}) concernant q_1, q_2 et q_3. Ces derniers dépendent des seuils de satisfaction fournis par S_1 (*i.e.* d_1 et d_2) et de ceux fournis par S_2 (*i.e.* d_3 et d_4). Vu qu'il s'agit de calculer la qualité minimale garantie, le seuil de satisfaction de $QoiS_{min}$ (S_{comp}) correspond alors au seuil minimal de satisfaction. En effet, la qualité q_1 est fournie avec le seuil d_1 par S_1 et avec le seuil d_3 par S_2. La qualité q_1 est ainsi fournie avec le seuil *Min* (d_1, d_3) par le service S_{comp}. Dans le cas de qualité fournie par un seul service composant, le seuil de satisfaction est celui fourni par le service composant lui-même. Par exemple q_2 est fournie uniquement par S_1 avec le seuil d_2, alors que q_3 est fournie uniquement par S_2 avec le seuil d_4. Le seuil minimal de satisfaction de S_{comp} pour ces qualités (*i.e.* q_2 et q_3) est donc d_2 et d_4. Ainsi, la $QoiS_{min}(S_{comp}) = \{<q_1, Min\ (d_1, d_3)>, <q_2, d_2>, <q_3, d_4>\}$. Le Tableau 16 précise le mode de calcul du seuil de satisfaction de $QoiS_{min}(S_{comp})$, en affectant des valeurs à d_1 et d_3. Le deuxième élément de la Doc 3 est calculé comme suit :

$$D = Min_{1 \le i \le n, 1 \le j \le m} \; d_{ij} \Big/ q_j$$

Tableau 16. Mode de calcul du seuil minimal de satisfaction

d_1	d_3	Min (d_1, d_3)
+	+	+
+	++	+
+	-	-
+	--	--
++	++	++
++	-	-
++	--	--
-	-	-
-	--	--
--	--	--

Le troisième élément de la Doc 3 consiste à définir les services composants fournissant *D* et *Q*. Une fois les qualités du service composite définies, ainsi que les seuils de satisfaction correspondant, il est intéressant d'indiquer les services composants, notés $\{\boldsymbol{S}_i\}$, qui sont à l'origine de cette qualité minimale : la satisfaction de *Q*, avec le seuil minimal *D*. La qualité minimale du service composite $\mathbf{S}_{comp}$ est définie comme suit : $QoiS_{min}$ ($\mathbf{S}_{comp}$) = {<q_1, *min* (d_1, d_3) >, <q_2, d_2 >, <q_3, d_4 >}. Cependant, celle-ci (*i.e.* la $QoiS_{min}$ ($\mathbf{S}_{comp}$)) ne précise pas les services composants offrant q_1, q_2 et q_3, *i.e.* que la $QoiS_{min}$ ($\mathbf{S}_{comp}$), telle que définie précédemment, n'informe pas sur qui (*i.e.* $\boldsymbol{S}_1$, $\boldsymbol{S}_2$ ou les deux) offre quoi (*i.e.* {<q_1, *min* (d_1, d_3) >, <q_2, d_2 >, <q_3, d_4) >}) comme qualité. Par conséquent, nous proposons d'indiquer les services $\boldsymbol{S}_1$, $\boldsymbol{S}_2$ ou les deux permettant de satisfaire partiellement les buts qualité, q_1, q_2 et q_3, avec respectivement les seuils de satisfaction *min* (d_1, d_3), d_2 et d_4. Ainsi, la $QoiS_{min}$ ($\mathbf{S}_{comp}$) = {<q_1, *min* (d_1, d_3), {$\boldsymbol{S}_1$, $\boldsymbol{S}_2$ *ou les deux*} >, <q_2, d_2, {$\boldsymbol{S}_1$} >, <q_3, d_4, {$\boldsymbol{S}_2$}>}. Le troisième élément de la Doc 3 est calculé comme suit :

$$\sigma = \bigcup_{i=1}^{n} \{S_i\}$$

Enfin, dans le cas d'un service composite, il s'agit d'appliquer la règle 1 pour calculer la $QoiS_{min}$:

$$Q = \bigcup_{j=1}^{m} \{q_j\} \qquad D = Min_{1\le i\le n, 1\le j\le m} \, d_{ij} \big/ q_j \qquad \sigma = \bigcup_{i=1}^{n} \{S_i\} \qquad \textit{règle 1}$$

EXEMPLE

Dans l'exemple du système d'approvisionnement de produits, nous considérons le service séquentiel $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ =●($\mathbf{S}_{Entrer\ les\ produits\ directement}$, $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$, $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$) correspondant à l'approvisionnement de produits en introduisant

directement les produits et en les contrôlant de façon continue.

Le service $\mathbf{S}_{Entrer\ les\ produits\ directement}$ ne contribue à aucune qualité, tandis que le service $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$ contribue à satisfaire partiellement la Fiabilité et le Coût avec un seuil de satisfaction respectif de « + » et de « - ». Enfin le service $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$ contribue à satisfaire partiellement la Fiabilité avec le seuil de satisfaction « + ». Les services atomiques $\mathbf{S}_{Entrer\ les\ produits\ directement}$, $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$ et $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$ ont des QoiS simples définies suivant la Doc 1 comme suit : $QoiS\ (\mathbf{S}_{Entrer\ les\ produits\ directement}) = \emptyset$, $QoiS\ (\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}) = \mathbf{S}_{Contrôler\ le\ stock\ en\ continu}.\{<Fiabilité,+>,<\ Coût\ ,->\}$ et $QoiS\ (\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}) = \mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}.\{<Fiabilité,+>\}$.

Par application de la règle 1, la $QoiS_{min}$ ($\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$) est calculée comme suit :

L'ensemble de buts qualité, *Q*, auxquels le service $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ contribue à satisfaire partiellement est : $Q = \{Fiabilité, Coût\}$;

Ensuite, pour chaque but qualité (*i.e. Fiabilité* et *Coût*), le seuil minimal de satisfaction, *D*, est calculé en observant les seuils offerts par les différents services composants offrant *Q*. Ainsi, $D = \{<Fiabilité, +>, <Coût, ->\}$ (*cf.* Tableau 16) ; et

Enfin, l'ensemble des services composants, σ, (*i.e.* $\mathbf{S}_{Entrer\ les\ produits\ directement}$, $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$ et $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$) contribuant à fournir ces qualités avec ce seuil minimal de satisfaction (*i.e.* $<Précision, +>$ *et* $<Coût, ->$) est indiqué. Par exemple, les services $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$ et $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$ contribuent à la qualité de *Précision* avec le seuil « + ». Donc, $\sigma = \{<Précision,+,\{\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}, \mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}\}>\}$. Par conséquent la $QoiS_{min}$ du service composite $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ est égale à :

$QoiS_{min}(\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}) = \{<Fiabilité, +, \{\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}, \mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}\}>, <Coût,-, \{\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}\}>\}$

La Figure 79 montre la représentation graphique de la $QoiS_{min}$ du service $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$.

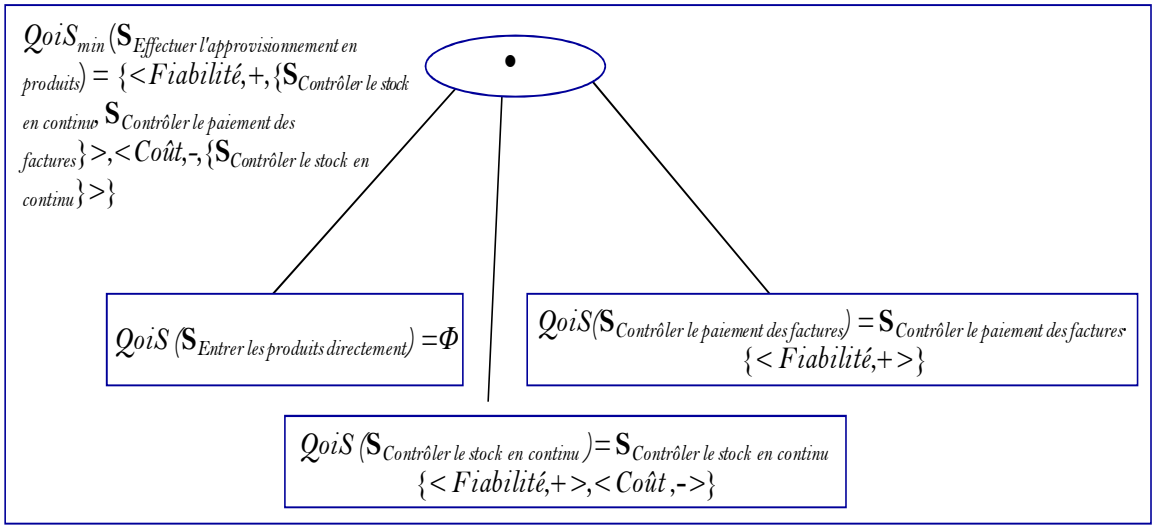

Figure 79. Représentation graphique de la $QoiS_{min}$ de $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$

6.3.2.2.3.3 *LA QOIS$_{MIN}$ VARIANTE*

Dans le cas d'un service variation (« $\otimes$ », « v », « $\cup$ »), nous adoptons, également, la Doc 3 pour représenter la $QoiS_{min}$ du service à variation $\boldsymbol{S}_{var}$. La Doc 3 exprime la qualité de service par l'ensemble de buts qualité que le service $\boldsymbol{S}_{var}$ (ou $\boldsymbol{S}_a$) contribue à satisfaire partiellement (*i.e. Q*) et les seuils minimaux (*i.e. D*) correspondant. Comme le montre la Doc 3, la $QoiS_{min}$ est indiquée par trois éléments. Pour expliquer les différents éléments de la Doc 3, nous considérons l'exemple d'un service à variation $\boldsymbol{S}_{var}$.

La première partie de la Doc 3 se base sur les principes de la théorie des ensembles. En fait, telle que définie plus haut, la qualité variante reflète un lien d'évaluation de type « *Ou* » entre les qualités composantes (*cf.* Section 6.3.2.2.2.2). Autrement dit, la qualité minimale variante est vérifiée, quelle que soit la variante empruntée. Ainsi, elle est définie comme l'intersection des buts qualité qui sont fournis par les différents services composants. Soient $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$ deux services composants du service à variation $\boldsymbol{S}_{var}$. Soient $A= \{q_1, q_2\}$ et $B= \{q_1, q_3\}$ les ensembles de buts qualité satisfaits partiellement par $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$. L'exécution de $\boldsymbol{S}_{var}$ implique l'exécution de l'un des services composants, à savoir $\boldsymbol{S}_1$ ou $\boldsymbol{S}_2$, dans le cas de « $\otimes$ » et « $\cup$ », et peut être l'exécution de tous les services composants, à savoir $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$, dans le cas de « v » (voir la sémantique des opérateurs d'agrégation au Chapitre 4). Dans ce dernier cas, la qualité minimale se comporte comme une $QoiS_{min}$ composite (*cf.* Section 6.3.2.2.3.2). L'exécution de $\boldsymbol{S}_1$ fournit la QoiS correspondant à *A* et l'exécution de $\boldsymbol{S}_2$ procure la QoiS correspondant à *B*. Ceci implique que le service agrégat à variation $\boldsymbol{S}_{var}$ va profiter de la qualité du $\boldsymbol{S}_1$ ou de celle de $\boldsymbol{S}_2$. Cependant, la qualité minimale du service à variation $\boldsymbol{S}_{var}$ est celle fournie simultanément par les services $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$, *i.e* A et B. Autrement dit, quel que soit le service

composant choisi $\boldsymbol{S}_1$ ou $\boldsymbol{S}_2$, la qualité minimale doit être satisfaite partiellement. Par conséquent, la QoiS minimale de $\boldsymbol{S}_{var}$ correspond à l'intersection des deux ensembles *A* et *B* (*i.e.* $A \cap B$). Ainsi, la QoiS_{min} ($\boldsymbol{S}_{var}$) = $\{q_1\}$. Ceci se traduit par :

$$\forall q_j, q_j \in A \cap B \equiv \left(\left(q_j \in A\right) \wedge \left(q_j \in B\right)\right)$$

Le premier élément de la Doc 3 correspond donc à l'intersection des buts qualité fournis par les services composants. Elle est calculée comme suit :

$$Q = \bigcap_{j=1}^{m} \{q_j\}$$

Le deuxième élément de la Doc 3 consiste à définir, pour chaque but qualité, le seuil minimal de satisfaction. Ce dernier est calculé de la même manière que celui de la QoiS_{min} composite (*cf.* Section 6.3.2.2.3.2). En effet, il s'agit de définir, pour chaque but qualité, le seuil garanti de satisfaction par le service à variation. Soient $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$ les services composants du service $\boldsymbol{S}_{var}$. Soient $A = \{<q_1, d_1>, <q_2, d_2>\}$ et $B = \{<q_1, d_3>, <q_3, d_4>\}$ les ensembles de buts qualité satisfaits partiellement par $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$, ainsi que les seuils de satisfaction correspondant. Les seuils de satisfaction d_1, d_2, d_3 et d_4 correspondent aux valeurs « ++ », « + », « - » ou « -- ». La QoiS minimale de $\boldsymbol{S}_{var}$ correspond à l'intersection des deux ensembles *A* et *B*, *i.e.* la QoiS_{min} ($\boldsymbol{S}_{var}$) = $\{q_1\}$. Cependant, $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$ fournissent q_1 avec les seuils de satisfaction respectifs d_1 et d_3. Par conséquent, il s'agit de définir le seuil de satisfaction de QoiS_{min} ($\boldsymbol{S}_{var}$). Vu que nous souhaitons calculer la qualité minimale, alors le seuil de satisfaction de QoiS_{min} ($\boldsymbol{S}_{var}$) correspond au seuil minimal fourni par les services composants. En effet, la qualité q_1 est fournie avec le seuil d_1 par $\boldsymbol{S}_1$ et avec le seuil d_3 par $\boldsymbol{S}_2$. La qualité q_1 est ainsi fournie avec le seuil *Min* (d_1, d_3) par $\boldsymbol{S}_{var}$. Elle est donc la qualité minimale de $\boldsymbol{S}_{var}$. Ainsi, la QoiS_{min} ($\boldsymbol{S}_{var}$) = $\{<q_1, Min\ (d_1, d_3)>\}$. Le Tableau 16 résume le mode de calcul du seuil minimal. Le deuxième élément de la Doc 3 est calculé comme suit :

$$D = Min_{1 \le i \le n, 1 \le j \le m} \, d_{ij} \Big/ q_j$$

Le troisième élément de la Doc 3 consiste, comme pour la QoiS_{min} composite (*cf.* Section 6.3.2.2.3.2), à définir les services composants, notés $\{\boldsymbol{S}_i\}$, fournissant *D* et *Q*. La qualité minimale du service à variation $\mathbf{S}_{\text{var}}$ est définie comme suit : QoiS_{min} ($\mathbf{S}_{\text{var}}$) = $\{<q_1, min\ (d_1, d_3)>\}$. Néanmoins, celle-ci (*i.e.* la QoiS_{min} ($\mathbf{S}_{\text{var}}$)) ne précise pas les services composants offrant q_1. Les services composants de $\mathbf{S}_{\text{var}}$ offrant cette qualité (*i.e.* q_1) avec ce seuil (*i.e.* *min* (d_1, d_3)) sont forcément tous les services (*i.e.* $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$) dans le cas de $d_1 = d_3$ et l'un des deux services (*i.e.* $\boldsymbol{S}_1$ ou $\boldsymbol{S}_2$) dans le cas de ($d_1 <= d_3$) ou ($d_1 >= d_3$). Par conséquent, la QoiS_{min} ($\mathbf{S}_{\text{var}}$)

= {<q_1, *min* (d_1, d_3), {$\boldsymbol{S}_1$*ou* $\boldsymbol{S}_2$ *ou les deux}* >}. Lorsqu'un service composant (*i.e.* $\boldsymbol{S}_1$ ou $\boldsymbol{S}_2$) du service à variation $\mathbf{S}_{var}$ ne fournit pas de qualité (*i.e.* QoiS($\boldsymbol{S}_1$)=Φ ou QoiS($\boldsymbol{S}_2$)=Φ), alors la $QoiS_{min}$ variante est vide également (*i.e.* $QoiS_{min}$ ($\mathbf{S}_{var}$)=Φ), car il n'existe pas de qualités communes fournies par tous les services composants de $\mathbf{S}_{var}$. Par conséquent, la qualité minimale indique ce que peut au minimum offrir un service, quel que soit la variante choisie. Cette qualité minimale est nécessaire à connaitre, lorsqu'il s'agit de sélectionner, parmi plusieurs services agrégats (de plusieurs fournisseurs) réalisant le même but fonctionnel, les plus adéquats (par rapport au contexte de qualité du client) (*cf.* Chapitre 8). De plus, à travers ses différentes configurations, le service agrégat peut offrir une meilleure qualité (*cf.* Chapitre 8). Le troisième élément de la Doc 3 est calculé comme suit :

$$\sigma = \bigcup_{i=1}^{n} \{S_i\}$$

Enfin, dans le cas d'un service à variation, il s'agit d'appliquer la règle 2 pour calculer la $QoiS_{min}$:

$$Q = \bigcap_{j=1}^{m} \{q_j\} \qquad D = Min_{1\le i\le n, 1\le j\le m} \, d_{ij} \big/ q_j \qquad \sigma = \bigcup_{i=1}^{n} \{S_i\} \qquad \textit{règle 2}$$

EXEMPLE

Dans l'exemple du système d'approvisionnement de produits, nous considérons le service à choix alternatif $\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$=⊗ ($\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$, $\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$) correspondant à l'acquisition de produits par planification, soit en surveillant le seuil de réapprovisionnement, soit en faisant des prévisions stratégiques.

Le service atomique $\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$ contribue partiellement à satisfaire la *Performance* et le *Coût* avec les seuils de satisfaction respectif de « + » et de « + », tandis que le service atomique $\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$ contribue partiellement aux mêmes qualités avec les seuils de satisfaction respectif de « ++ » et de « - ». Les services $\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$ et $\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$ ont des qualités définies suivant la Doc 1 comme suit : *QoiS* ($\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$) = $\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$.{< *Performance*,+>,< *Coût* ,+>} et *QoiS* ($\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$) = $\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$. {< *Performance*, ++>, < *Coût*, ->}.

Par application de la règle 2, la $QoiS_{min}$($\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$) est calculée comme suit :

L'ensemble minimal de buts qualité, *Q*, auxquels le service $\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$ contribue à satisfaire partiellement est : *Q* = {*Performance*, *Coût*} ;

Ensuite, pour chaque but qualité (*i.e.* *Performance* et *Coût*), le seuil de satisfaction minimal, *D*, est calculé en observant les seuils offerts par chaque variante. Ainsi, *D* = {< *Performance, +*>, <*Coût, ->} (*cf.* Tableau 16), car toute les variantes de ce service offrent au minimum ce seuil ; et

Enfin, l'ensemble des services composants, σ, contribuant à fournir les qualités *Performance* et *Coût* avec les seuils respectifs « + » et « - » (*i.e.* {< *Performance,+*>,<*Coût,->*}) est indiqué. Le service $\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$ contribue à la qualité de *Performance* avec le seuil « + », tandis que le service $\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$ contribue à la qualité de *Performance* avec le seuil « ++ ». Ainsi, σ = {<*Performance,+,{*$\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$*}*>}. Par conséquent la $QoiS_{min}$ du service à variation $\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$ est :

$QoiS_{min}$ ($\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$) = {<*Performance,+,{*$\mathbf{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$*}*>,<*Coût,-,{*$\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$ *}*>}

La Figure 80 montre la représentation graphique de la $QoiS_{min}$ du service $\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$.

Nous avons présenté précédemment la qualité associée au service intentionnel. Celle-ci peut être une QoiS simple lorsqu'il s'agit d'un service atomique et une QoiS globale lorsqu'il s'agit d'un service agrégat. Pour cette dernière (*i.e.* la QoiS globale), nous avons préféré calculer une qualité minimale afin de ne pas surestimer la qualité des services intentionnels. La qualité minimale ($QoiS_{min}$) est indiquée afin de préciser ce que peut au minimum offrir un service, en termes de buts qualité, ainsi que de seuils de satisfaction. La $QoiS_{min}$ peut être une $QoiS_{min}$ composite, lorsqu'il s'agit d'un service composite, ou une $QoiS_{min}$ variante, lorsqu'il s'agit d'un service à variation.

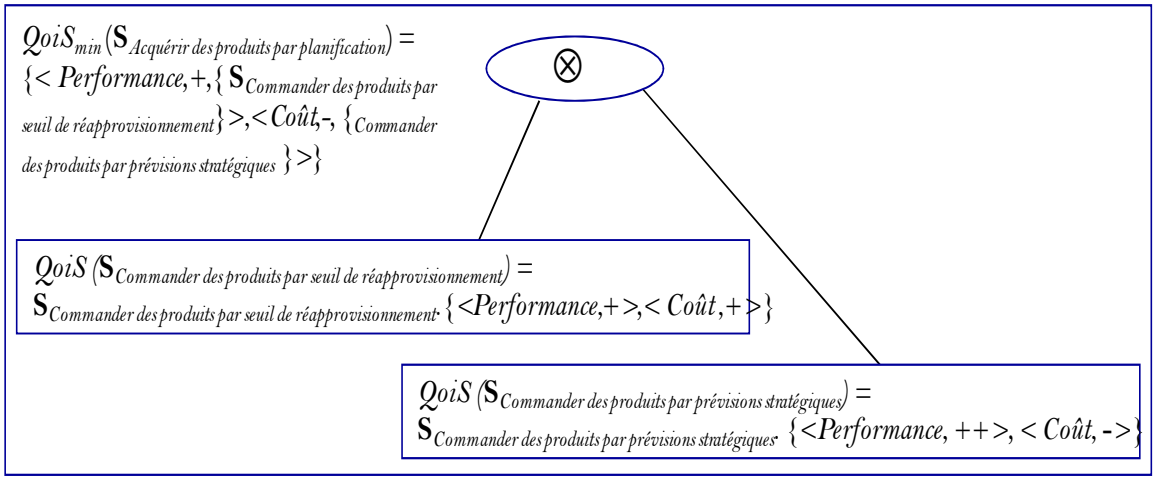

Figure 80. Représentation graphique de la $QoiS_{min}$ de $\mathbf{S}_{Acquérir\ des\ produits\ par\ planification}$

Nous précisions que la qualité minimale nous permet d'une part de préciser la qualité garantie par le service, et d'autre part de sélectionner, parmi plusieurs services agrégats (de plusieurs fournisseurs) réalisant le même but, les services les plus adéquats, en fonction du contexte qualité du client (*cf.* Chapitre 8).

Néanmoins, cette qualité minimale ne peut être utilisé pour comparer les configurations d'une même service agrégat, car elle met sur un pied d'égalité des configurations ayant beaucoup de « + » et un seul « - », et celles qui possèdent beaucoup de « - » et un seul « + ». Afin de remédier à ce problème, nous proposons de calculer la qualité moyenne, dite la $QoiS_{moy}$, qui est calculée dynamiquement par le prototype TOPSiQC, chaque fois qu'une configuration exécutable est générée. Nous présentons, dans la prochaine section, la définition et le calcul de la $QoiS_{moy}$.

6.3.2.2.4 LA QUALITE MOYENNE : $QOIS_{MOY}$

La qualité minimale $QoiS_{min}$ indique la qualité garantie par le service agrégat, sachant que ce dernier pourrait fournir une meilleure qualité en fonction du choix de ses configurations (*cf.* Chapitre 8). La $QoiS_{min}$ est exploitée par le processus de sélection de services, afin de choisir parmi plusieurs services (*cf.* Chapitre 8). Par exemple, la $QoiS_{min}$ du service $\boldsymbol{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ est définie comme suit : $QoiS_{min}$ $(\boldsymbol{S}_{Effectuer\ l'approvisionnement\ en\ produits}) = \{<Performance,-,\{\mathbf{S}_{Commander\ des\ produits\ manuellement}\}>,<Fiabilité,-,\{\mathbf{S}^*_{Etablir\ un\ inventaire\ par\ échantillonnage}\}>\}$. Toutefois, les configurations de ce service peuvent fournir de meilleures qualités, car il existe des services qui offrent de très bonnes qualités, tels que $\mathbf{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$ (avec une <Performance,++>) et $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$ (avec une <Fiabilité,++>). La $QoiS_{min}$ privilégie la garantie d'un seuil de satisfaction, par une agrégation intentionnelle de service. Par conséquent, la propagation des seuils « - » et « -- »

est privilégié au détriment des seuils plus satisfaisant, à savoir « + » et « ++ ». Nous proposons alors de comparer les configurations du service agrégat, non plus sur la base de la qualité minimale, mais en utilisant une qualité moyenne.

La qualité moyenne $QoiS_{moy}$ est alors calculée pour comparer les différentes configurations du service agrégat. Le service agrégat correspond à une agrégation de services composites et de services à variation (*cf.* Chapitre 4). La génération des différentes configurations du service agrégat correspond alors aux choix successifs de variantes. Chaque configuration du service agrégat, désignée par le code $\boldsymbol{Conf}_{i/i=1..m}$, correspond à un service composite séquentiel (*i.e.* un chemin).

La génération des configurations du service agrégat dépend de ses services composants. Le Tableau 17 résume le nombre des configurations généré, lequel dépend du type de service. Dans le cas de service à choix alternatif ou de service à variation de chemin, le nombre de configurations correspond au nombre de variantes de ces services. Par exemple, à partir du service $\otimes(\mathbf{S}_1, \mathbf{S}_2, \mathbf{S}_3)$, nous pouvons générer trois configurations, $\mathbf{S}_1$, $\mathbf{S}_2$, ou $\mathbf{S}_3$. Pour le service à choix, le nombre de configurations est égal à $(2^n\text{-}1)$; *n* étant le nombre de variantes. Par exemple, à partir du service $v(\mathbf{S}_1, \mathbf{S}_2)$, nous pouvons générer trois configurations, $\mathbf{S}_1$, $\mathbf{S}_2$, ou ● ($\mathbf{S}_1$, $\mathbf{S}_2$). Enfin, si le service composant est un service séquentiel, alors le nombre de configurations correspond au service lui-même, car ce dernier ne comprend pas de la variation.

Tableau 17. Nombre de configurations selon le type de service à variation

Type service	**Nombre configurations générées**
Service à choix alternatif ($\otimes$): $\otimes(\mathbf{S}_1, \mathbf{S}_2, ...\mathbf{S}_n)$	« n » configurations
Service à variation de chemin($\cup$): $\cup(\mathbf{S}_1, \mathbf{S}_2, ... \mathbf{S}_n)$	
Service à choix multiple (v) : $v(\mathbf{S}_1, \mathbf{S}_2, ... \mathbf{S}_n)$	« 2^n-1 » configurations
Service séquentiel (●) : ●($\mathbf{S}_1, \mathbf{S}_2, ... \mathbf{S}_n$)	1 configuration

Ainsi, soit *m* le nombre de configurations, l'indice *i* indique le numéro de la configuration, relatif au numéro du service agrégat. Ces configurations générées doivent être exécutables ; elles doivent satisfaire les contraintes de dépendance existantes entre les variantes du service agrégat (*cf.* Section 6.3.1). Une configuration $\boldsymbol{Conf}_i$ est dite exécutable lorsque toutes les contraintes, associées au service agrégat, sont satisfaites (*cf.* Section 6.3.1.4).

Une fois les configurations exécutables générées, il s'agit alors de calculer la $QoiS_{moy}$ de chacune d'elles. Nous adoptons la Doc 4 pour représenter la $QoiS_{moy}$ de la configuration $\boldsymbol{Conf}_i$:

$$QoiS_{moy}\ (\boldsymbol{Conf}_i) = \{<Q, D>\} \qquad \textit{Doc 4}$$

Q correspond à l'ensemble des buts qualités (q_j) que la configuration $\boldsymbol{Conf}_i$ contribue partiellement à satisfaire ; *D* correspond aux seuils de satisfaction correspondant à chaque but qualité.

Le premier élément de la Doc 4 (*i.e. Q)* se base sur les principes de la théorie des ensembles. Comme la QoiS composite (*cf.* Section 6.3.2.2.2.1), les q_j d'une configuration reflète un lien d'évaluation de type « *Et* » entre les q_j de ses services composants. Soient $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$ deux services composants de la configuration $\boldsymbol{Conf}_1$. Soient $A= \{q_1, q_2\}$ et $B= \{q_1, q_3\}$ les ensembles de buts qualité satisfaits partiellement par $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$, l'exécution de $\boldsymbol{Conf}_1$ implique l'exécution de tous les services composants, à savoir $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$. L'exécution de $\boldsymbol{S}_1$ fournit la QoiS correspondant à *A* et l'exécution de $\boldsymbol{S}_2$ procure la QoiS correspondant à *B*. Ceci implique que la configuration $\boldsymbol{Conf}_1$ va bénéficier des qualités des services composants $\boldsymbol{S}_1$ « *Et* » $\boldsymbol{S}_2$. Par conséquent, la $QoiS_{moy}$ de $\boldsymbol{Conf}_1$ correspond à l'union des deux ensembles *A* et *B* (*i.e. A U B*). Ainsi, la $QoiS_{moy}$ ($\boldsymbol{Conf}_1$) = $\{q_1, q_2, q_3\}$. Ceci se traduit par :

$$\forall q_j, q_j \in A \cup B \equiv \left(\left(q_j \in A\right) \vee \left(q_j \in B\right)\right)$$

Le premier élément de la Doc 4 correspond donc à la réunion des buts qualité fournis par les services composants de la configuration. Elle est calculée comme suit :

$$Q = \bigcup_{j=1}^{m} \left\{q_j\right\}$$

Le deuxième élément de la Doc 4 consiste à définir, pour chaque but qualité, le seuil moyen de satisfaction. Soient $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$ les services composants de la configuration $\boldsymbol{Conf}_1$. Soient $A= \{q_1, q_2\}$ et $B= \{q_1, q_3\}$ les ensembles de buts qualité satisfaits partiellement par $\boldsymbol{S}_1$ et $\boldsymbol{S}_2$. La $QoiS_{moy}$ de $\boldsymbol{Conf}_1$ correspond à l'union des deux ensembles *A* et *B, i.e.* la $QoiS_{moy}$ ($\boldsymbol{Conf}_1$) = $\{q_1, q_2, q_3\}$. Cependant, $\boldsymbol{S}_1$ fournit q_1 *et* q_2 avec les seuils de satisfaction respectifs d_1 et d_2. De façon identique, $\boldsymbol{S}_2$ fournit q_1 et q_3 avec les seuils de satisfaction respectifs d_3 et d_4. Aussi, $A= \{<q_1, d_1>, <q_2, d_2>\}$ et $B= \{<q_1, d_3>, <q_3, d_4>\}$. Les seuils de satisfaction d_1, d_2, d_3 et d_4 correspondent aux valeurs « ++ », « + », « *?* » , « - » ou « -- ». Ces seuils de satisfaction correspondent respectivement à « 2 », « 1 », « 0 », « -1 », et « -2 » (voir le Chapitre 8 pour plus de détail). Il s'agit alors de définir les seuils de satisfaction de $QoiS_{moy}$ ($\boldsymbol{Conf}_1$) concernant q_1, q_2 et q_3. Ces derniers dépendent des seuils de satisfaction fournis par $\boldsymbol{S}_1$ (*i.e.* d_1

et d_2) et de ceux fournis par $\boldsymbol{S_2}$ (*i.e.* d_3 et d_4). Vu qu'il s'agit de calculer la qualité moyenne, le seuil de satisfaction de $QoiS_{moy}$ ($\boldsymbol{Conf_1}$) correspond alors au seuil moyen de satisfaction. Le seuil moyen correspond alors à la somme des seuils significatifs (*i.e.* seuil ≠ ?), divisé par le nombre de ces seuils. En effet, la qualité q_1 est fournie avec le seuil d_1 par S_1 et avec le seuil d_3 par S_2. La qualité q_1 est alors fournie avec le seuil moyen $((d_1 + d_3)/2)$ pour la configuration $\boldsymbol{Conf_1}$. Dans le cas de qualité fournie par un seul service composant, le seuil de satisfaction est celui fourni par le service composant lui-même. Par exemple q_2 est fournie uniquement par S_1 avec le seuil d_2, alors que q_3 est fournie uniquement par $\boldsymbol{S_2}$ avec le seuil d_4. Le seuil moyen de satisfaction de $\boldsymbol{Conf_1}$ pour ces qualités (*i.e.* q_2 et q_3) est donc d_2 et d_4. Ainsi, la $QoiS_{moy}(\boldsymbol{Conf_1}) = \{<q_1, (d_1 + d_3)/2>, <q_2, d_2>, <q_3, d_4>\}$. Le deuxième élément de la Doc 4 est calculé comme suit :

$$\frac{\sum (d_{ij} \neq "?")}{nombre(d_{ij} \neq "?")} \Big/ q_j$$

Enfin, la $QoiS_{moy}$ est calculée comme suit :

$$Q = \bigcup_{j=1}^{m} \{q_j\} \qquad \frac{\sum (d_{ij} \neq "?")}{nombre(d_{ij} \neq "?")} \Big/ q_j$$

EXEMPLE

Dans l'exemple du système d'approvisionnement de produits, nous considérons la configuration $\mathbf{Conf}_{Effectuer\ l'approvisionnement\ en\ produits1} = \bullet(\mathbf{S}_{Entrer\ les\ produits\ directement}, \mathbf{S}_{Contrôler\ le\ stock\ par\ échantillonnage}, \mathbf{S}_{Contrôler\ le\ paiement\ des\ factures})$ correspondant à l'approvisionnement de produits en introduisant directement les produits et en les contrôlant par échantillonnage.

Le service $\mathbf{S}_{Entrer\ les\ produits\ directement}$ ne contribue à aucune qualité, tandis que le service $\mathbf{S}_{Contrôler\ le\ stock\ par\ échantillonnage}$ contribue à satisfaire partiellement la Fiabilité avec un seuil de satisfaction « - ». Enfin le service $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$ contribue à satisfaire partiellement la Fiabilité avec le seuil de satisfaction « ++ ». Ces seuils de satisfaction correspondent respectivement à « -1 » et « 2 ». Les services atomiques $\mathbf{S}_{Entrer\ les\ produits\ directement}$, $\mathbf{S}_{Contrôler\ le\ stock\ par\ échantillonnage}$ et $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$ ont des QoiS simples définies suivant la Doc 1 comme suit : $QoiS(\mathbf{S}_{Entrer\ les\ produits\ directement}) = Ø$, $QoiS(\mathbf{S}_{Contrôler\ le\ stock\ par\ échantillonnage}) = \mathbf{S}_{Contrôler\ le\ stock\ par\ échantillonnage}.\{<Fiabilité,->\}$ et $QoiS(\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}) = \mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}.\{<Fiabilité,++>\}$.

Par application de la Doc 4, la $QoiS_{moy}$ ($\mathbf{Conf}_{Effectuer\ l'approvisionnement\ en\ produits1}$) est calculée comme suit :

L'ensemble de buts qualité, *Q*, auxquels la configuration **Conf**$_{Effectuer\ l'approvisionnement\ en\ produits1}$ contribue à satisfaire partiellement est : *Q* = {*Fiabilité*}, et pour chaque but qualité (*i.e. Fiabilité*), le seuil moyen de satisfaction, *D*, est calculé en observant les seuils offerts par les différents services composants offrant *Q*. Ainsi, *D* = {<*Fiabilité, 0.5*>}. Par conséquent la $QoiS_{moy}$ de la configuration **Conf**$_{Effectuer\ l'approvisionnement\ en\ produits1}$ est égale à :

$$QoiS_{moy}(\mathbf{Conf}_{Effectuer\ l'approvisionnement\ en\ produits1}) = \{< Fiabilité, 0.5>\}$$

Nous avons présenté dans cette section la QoiS fournie par le service intentionnel. Celle-ci est définie comme étant l'ensemble des buts qualité que le service permet de satisfaire partiellement. La QoiS d'un service atomique est dite simple, tandis que la QoiS d'un service agrégat est dite globale. Une $QoiS_{min}$ est aussi calculée pour informer le client de la qualité minimale que peut garantir un service agrégat. La $QoiS_{min}$ est utilisée pour comparer les différents services agrégats. Pour comparer les configurations d'un même service agrégat, nous proposons de calculer la qualité moyenne $QoiS_{moy}$ de chaque configuration exécutable.

Afin de mettre à disposition des clients les contraintes de dépendance entre services, ainsi que leurs qualités, nous présentons, dans la prochaine section, la publication des contraintes et de la QoiS du service intentionnel dans le registre des services intentionnels. La $QoiS_{moy}$ n'est pas publiée, car elle est calculée dynamiquement pour chaque configurations générée.

6.3.3 PUBLICATION DES CONTRAINTES DE DEPENDANCE ET DE LA QOIS DU SERVICE INTENTIONNEL

Le service intentionnel doit être disponible dans l'annuaire des services afin de permettre la localisation (ou la découverte) des services (atomiques et agrégats). La localisation des services est également orientée but. Ainsi, à partir du but que le client souhaite réaliser, l'annuaire des services met à la disposition du client les services permettant de réaliser le but exprimé par le client. Particulièrement, les buts des clients peuvent concerner aussi bien des buts fonctionnels que des buts qualités, telles que la sécurité, la performance ou la précision. Par conséquent, l'annuaire des services doit, également, mettre à la disposition des clients, les qualités que les services permettent de satisfaire, afin de mieux classer les services. De plus, il peut exister des contraintes de dépendance (Exige ou Exclut) entre les services qu'il est important de préciser, afin de valider l'exécutabilité des configurations d'un service agrégat.

L'utilisation intensive du langage standard XML (*eXtensible Markup Language*) constitue l'une des caractéristiques fondamentales de la technologie des services Web (Kellert&Toumani,2003). Par conséquent, nous proposons de se baser sur le langage XML

afin de publier les contraintes et la QoiS du service intentionnel dans l'annuaire en traduisant le modèle *MiS-q* vers le langage XML.

Dans ce rapport, nous proposons une publication des contraintes de dépendance et celle du service intentionnel et de sa qualité.

La Figure 81 montre une description des contraintes de dépendance entre services. Comme le montre la Figure 81, la contrainte de dépendance (*Dependance* à la ligne 8 de la Figure 81) concerne un service agrégat (ligne 10 de la Figure 81) et elle est définie comme un ensemble de règles (*Rule* à la ligne 12 de la Figure 81), lesquelles peuvent être de type Exige (*Imply* à la ligne 19 de la Figure 81) ou Exclut (*Exclude* à la ligne 20 de la Figure 81). Les règles d'exigence et d'exclusion concernent un service cible (*Target* à la lignes 27 et 33 de la Figure 81) et un ou plusieurs services sources (*Source* à la ligne 26 et 32 de la Figure 81). La publication des contraintes est effectuée lors de la publication du service agrégat, car elles permettent de valider l'exécutabilité des configurations de ce service.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3      targetNamespace="http://xml.netbeans.org/schema/Dependence"
4      xmlns:tns="http://xml.netbeans.org/schema/Dependence"
5      elementFormDefault="qualified">
6
7  <xsd:element name="Dependence">
8     <xsd:complexType>
9       <xsd:element ref="Aggregate Service" />
10      <xsd:sequence>
11        <xsd:group ref="tns:Rule" minOccurs="0" maxOccurs="unbounded"/>
12      </xsd:sequence>
13    </xsd:complexType>
14  </xsd:element>
15
16  <xsd:group name="Rule">
17         <xsd:choice>
18                 <xsd:element ref="tns:Imply" />
19                 <xsd:element ref="tns:Exclude" />
20         </xsd:choice>
21  </xsd:group>
22
23  <xsd:element name="Imply">
24    <xsd:complexType>
25        <xsd:attribute name="source" type="xsd:string" use="required" />
26        <xsd:attribute name="target" type="xsd:string" use="required" />
27    </xsd:complexType>
28  </xsd:element>
29  <xsd:element name="Exclude">
30    <xsd:complexType>
31        <xsd:attribute name="source" type="xsd:string" use="required" />
32        <xsd:attribute name="target" type="xsd:string" use="required" />
33         </xsd:complexType>
34  </xsd:element>
35 </xsd:schema>
```

Figure 81. La description des contraintes de dépendance

La Figure 82 met en évidence la description du service intentionnel et de sa qualité, laquelle peut être publiée par le fournisseur métier. Comme le montre la Figure 82, le service intentionnel est représenté par un groupe de services (*ServicesChild* à la ligne 9 de la Figure 82). Celui-ci peut être un service atomique (*Leaf* à la ligne 20 de la Figure 82) ou un service agrégat. Ce dernier peut être un service séquentiel (*And* à la ligne 17 de la Figure 82) ; un service à variation de chemin (*Xor* à la ligne 18 de la Figure 82) ; un service à choix multiple (*Or* à la ligne 19 de la Figure 82) ; ou un service à choix alternatif (*LXor* à la ligne 21 de la Figure 82) . Un service atomique (*Leaf* à la ligne 27 de la Figure 82) est défini par un identifiant (*Id* à la ligne 32 de la Figure 82), et il peut fournir des qualités (*QoS* à la ligne 30 de la Figure 82). La qualité du service atomique est décrite par le but qualité fourni par ce service (Q*uality* à la ligne 39 de la Figure 82), ainsi que le seuil de satisfaction (D*egree* à la ligne 40 de la Figure 82) correspondant. Le seuil de satisfaction peut être un « + » (ligne 13 de la Figure 87), un « ++ » (ligne 14 de la Figure 87), un « - » (ligne 15 de la Figure 87) ou « -- » (ligne 16 de la Figure 87), lesquels correspondent respectivement aux expressions « Satisfait», « Très satisfait», « Peu satisfait » et « Pas du tout satisfait ». Une description complète du service intentionnel est présentée à l'Annexe 4.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3      targetNamespace="http://xml.netbeans.org/schema/Tree"
4      xmlns:tns="http://xml.netbeans.org/schema/Tree"
5   elementFormDefault="qualified" >
6
7   <xsd:element name="Services">
8     <xsd:complexType>
9         <xsd:group ref="tns:ServicesChild" minOccurs="1" maxOccurs="1"/>
10    </xsd:complexType>
11  </xsd:element>
12
13  …..
14
15       <xsd:group name="ServicesChild">
16          <xsd:choice>
17                  <xsd:element ref="tns:And" />
18                  <xsd:element ref="tns:Xor" />
19                  <xsd:element ref="tns:Or" />
20                  <xsd:element ref="tns:Leaf" />
21                  <xsd:element ref="tns:LXor" />
22           </xsd:choice>
23      </xsd:group>
24
25  ….
26
27    <xsd:element name="Leaf">
28          <xsd:complexType>
29            <xsd:sequence>
30    <xsd:element ref="tns:QoS" minOccurs="0" maxOccurs="unbounded"/>
31              </xsd:sequence>
32           <xsd:attribute name="id" use="required" type="xsd:string" />
33          </xsd:complexType>
34      </xsd:element>
35
36      <xsd:element name="QoS">
37          <xsd:complexType>
38              <xsd:sequence>
39       <xsd:element ref="tns:Quality" minOccurs="1" maxOccurs="1"/>
40       <xsd:element ref="tns:Degree" minOccurs="1" maxOccurs="1"/>
41              </xsd:sequence>
42          </xsd:complexType>
43      </xsd:element>
44
45  <xs:element name ="Quality" type="xsd:string"/>
46  <xsd:element name="Degree" type="xsd:string">
47   <xsd:choice>
48      <xsd:element  name="+" />
49      <xsd:element name="++" />
70      <xsd:element name="-" />
71      <xsd:element name="--" />
72   </xsd:choice>
73  </xsd:element>
74
75  </xsd:schema>
```

Figure 82. La description du service intentionnel

EXEMPLES

Dans l'exemple de l'approvisionnement de produits, nous considérons le service séquentiel $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits} = \bullet($ $\mathbf{S}_{Entrer\ les\ produits\ directement}$, $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$, $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures})$, défini plus haut (*cf.* Section 6.3.2.2.3.2). Nous proposons de publier la qualité des services atomiques $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$ et $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$. Les Figure 83 et Figure 84 illustrent respectivement la QoiS des services $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$ et $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$.

Comme présenté dans les Figure 83 et Figure 84, la QoiS des services atomiques est définie en fonction des buts qualité. Pour le service agrégat, la QoiS globale ou la $QoiS_{min}$ est calculée pour définir la qualité du service. Par exemple, il est possible de calculer la qualité du service $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produit}$. En effet, étant donné qu'il s'agit d'un service composite, alors on applique la *règle 1* et la Doc 3 (*cf.* Section 6.3.2.2.3) pour calculer et documenter la qualité minimale de $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$. Ainsi, la $QoiS_{min}$ $(\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}) = \{<Fiabilité,+,\{\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}, \mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}\}>,<Coût,-,\{\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}\}>\}$. Également, il est possible de documenter la QoiS globale de $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produit}$, en appliquant la *Doc 2* (*cf.* Section 6.3.2.2.3.2).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Service>
3   <tree:Leaf id= "S Contrôler le stock en continu">
4
5
6   <QoS>
7    <quality> Fiabilité </quality>
8    <degree> + </degree>
9   </QoS>
10  <QoS>
11   <quality> Coût </quality>
12   <degree> - </degree>
13  </QoS>
14
15  </Service>
```

Figure 83. La QoiS du service atomique $\mathbf{S}_{Contrôler\ le\ stock\ en\ continu}$

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Service>
3   <tree:Leaf id= "S Contrôler le paiement des factures">
4
5   <QoS>
6    <quality> Fiabilité </quality>
7    <degree> + </degree>
8   </QoS>
9
10  </Service>
```

Figure 84. La QoiS du service atomique $\mathbf{S}_{Contrôler\ le\ paiement\ des\ factures}$

La Figure 85 montre une description succincte du service agrégat séquentiel $\mathbf{S}_{\text{Réserver un billet de train}}=\bullet(\otimes(\mathbf{S}_{\text{Réserver un billet de train par internet}},\mathbf{S}_{\text{Réserver un billet de train en agence}}),\mathbf{S}^{*}_{\text{Réserver un billet de train réduit}},\cup(\mathbf{S}_{\text{Annuler un billet de train}},\bullet(\otimes(\mathbf{S}_{\text{Payer billet par carte bancaire}},\mathbf{S}_{\text{Payer billet par chèque}},\mathbf{S}_{\text{Payer billet par espèce}}),\nu(\mathbf{S}_{\text{Présenter une carte de réduction}},\mathbf{S}_{\text{Présenter billet de train}})))$. La description complète de ce service est présentée à l'Annexe 4. La qualité de tous les services atomiques du service $\mathbf{S}_{\text{Réserver un billet de train}}$ est définie à la Figure 85. Par exemple, le service $\mathbf{S}_{\text{Réserver un billet de train par internet}}$ fournit la qualité de commodité avec le seul « + » (lignes 8-13 à la Figure 85). Au service agrégat $\mathbf{S}_{\text{Réserver un billet de train}}$ sont associées une contrainte d'exigence (ligne 14 à la Figure 86) et deux contraintes d'exclusion (lignes 10-13 à la Figure 86). Ces contraintes doivent être vérifiées par les différentes configurations du service $\mathbf{S}_{\text{Réserver un billet de train}}$.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3        xmlns:tree='http://RéserverBilletDeTrain/Tree'
4        xsi:schemaLocation='http://RéserverBilletDeTrain/Tree Tree.xsd'>
5
6  <tree:And id="and-01">
7      <tree:LXor id="lxor-01">
8           <tree:Leaf id= "S Réserver un billet de train par internet">
9                 <tree:QoS>
10                      <tree:Quality>commodité</tree:Quality>
11                       <tree:Degree>+</tree:Degree>
12                </tree:QoS>
13           </tree:Leaf>
14             <tree:Leaf id= "S Réserver un billet de train en agence">
15                <tree:QoS>
16                      <tree:Quality>commodité</tree:Quality>
17                       <tree:Degree>-</tree:Degree>
18                </tree:QoS>
19           </tree:Leaf>
20    </tree:LXor>
21    <tree:Star id="star-01">
22         <tree:Leaf id= "S Réserver un billet de train réduit">
23                <tree:QoS>
24                      <tree:Quality>cout</tree:Quality>
25                       <tree:Degree>+</tree:Degree>
26                </tree:QoS>
27           </tree:Leaf>
28    </tree:Star>
29 …...
30
31 </tree:Services>
```

Figure 85. La description du service agrégat $\mathbf{S}_{\text{Réserver un billet de train}}$

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <dependence:Dependence xmlns:xsi='http://www.w3.org/2001/XMLSchema-
4  instance'
5        xmlns:dependence=' http://Assia/Dependence'
6        xsi:schemaLocation='http://Assia/Dependence Dependence.xsd'>
7
8  <aggregate service> Ref="S Réserver un billet de train" </aggregate service>
9
```

```
10  <dependence:Exclude source="S Réserver un billet de train par internet" target="S Payer un
11  billet de train par espèce"/>
12  <dependence:Exclude source="S Réserver un billet de train par internet" target="S Payer un
13  billet de train par chèque"/>
14  <dependence:Imply source="S Réserver un billet de train réduit" target=" S Présenter une
15  carte de réduction"/>
16  </dependence:Dependence>
```

Figure 86. Les contraintes de dépendance du service agrégat $S_{\text{Réserver un billet de train}}$

La Figure 87 met en évidence la publication des services agrégats dans un annuaire de services, ainsi que leurs qualités. En effet, l'opérateur *Xor* (ligne 6 de la Figure 87) exprime le choix entre plusieurs services agrégats. Ces derniers sont définis par des *Leaf* (ligne 7, ligne 13 et ligne 19 à la Figure 87), et ils fournissent une qualité minimale (*QoS* à la Figure 87) correspondant aux buts qualité (*Quality* à la ligne 9, ligne 15 et ligne 21 à la Figure 87) et les seuils minimaux de satisfaction (*Degree* à la ligne 10, ligne 16 et ligne 22 à Figure 87).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
3         xmlns:tree='http://RéserverBilletDeTrain/Tree'
4         xsi:schemaLocation='http://RéserverBilletDeTrain/Tree Tree.xsd'>
5
6   <tree:Xor id="xor-01">
7             <tree:Leaf id= "S1">
8                   <tree:QoS>
9                         <tree:Quality>commodité</tree:Quality>
10                         <tree:Degree>+</tree:Degree>
11                  </tree:QoS>
12            </tree:Leaf>
13            <tree:Leaf id= "S2">
14                  <tree:QoS>
15                        <tree:Quality>commodité</tree:Quality>
16                        <tree:Degree>-</tree:Degree>
17                  </tree:QoS>
18            </tree:Leaf>
19            <tree:Leaf id= "S3">
20                  <tree:QoS>
21                        <tree:Quality>cout</tree:Quality>
22                        <tree:Degree>+</tree:Degree>
23                  </tree:QoS>
24            </tree:Leaf>
25  …...
26  </tree:Xor>
27  </tree:Services>
```

Figure 87. Exemple de publication des services agrégats et leurs qualités

6.4 *CONCLUSION*

Nous avons présenté dans ce chapitre le modèle *MiS-q*, qui permet la représentation des contraintes de dépendance entre services, et la qualité fournie par le service intentionnel (*QoiS-Quality of intentional Service*), d'une manière orientée but et utilisateur final.

Le service intentionnel agrégat encapsule plusieurs points de variation. Les contraintes de dépendance sont alors utilisées afin d'exprimer des informations complémentaires aux opérateurs d'agrégation. Le modèle *MiS-q* décrit des contraintes d'exclusion (« Exclut ») ou d'inclusion (« Exige »). Elles précisent que le choix d'une variante donnée a une influence (d'exigence ou d'exclusion) sur les choix suivants des variantes. Les contraintes sont définies dans le contexte d'un service agrégat, et elles permettent de valider l'exécutabilité de ses configurations.

Le modèle *MiS-q* définit la QoiS comme étant l'ensemble des buts qualité, que les services contribuent partiellement à satisfaire. Il propose également un mode de calcul et de documentation permettant respectivement d'évaluer et de documenter la QoiS simple associé au service atomique, la QoiS globale définie pour un service agrégat et la $QoiS_{min}$ que le service intentionnel peut fournir au minimum. La qualité du service intentionnel est mise à la disposition des agents métier grâce à leurs publications. Pour comparer les configurations du service agrégat la $QoiS_{moy}$ est calculée.

Enfin, le modèle *MiS-q* introduit de la variabilité qualitative dans la modélisation des services intentionnels. En effet, les services, réalisant le même but, permettent de satisfaire partiellement différents buts qualité avec des seuils de satisfaction différents. Le service à variation et la QoiS variante introduisent ainsi de la flexibilité dans la manière de réaliser le but du service et la satisfaction partielle des buts qualités. Ceci apporte une flexibilité dans la composition de services, en permettant de réponde au mieux aux exigences fonctionnelles et non-fonctionnelles des agents métier.

CHAPITRE 7 : METHODOLOGIE DE PUBLICATION DE MiS-q

7.1 *INTRODUCTION*

Ce chapitre permet de publier les services intentionnels selon le modèle *MiS* étendu, appelé *MiS-q*, en prenant en compte :

- la qualité au niveau des services intentionnels, dite la QoiS *(Quality of intentional Service*) ; et
- les contraintes de dépendance entre les services représentant une variabilité métier dans une agrégation.

Nous proposons dans ce chapitre de réutiliser l'approche méthodologique de (Kaabi,2007) pour identifier et décrire les services intentionnels, et l'étendre à la prise en compte des deux éléments cités ci-dessus (*i.e.* QoiS et contrainte de dépendance).

Ce chapitre est composé de plusieurs sections. La section 2 présente la vue globale de la méthodologie de publication du modèle *MiS-q*. La section 3 présente l'intégration de la qualité à la carte des besoins. La section 4 définit la génération de la QoiS simple. La section 5 présente l'opérationnalisation des services atomiques. La section 6 définit la génération de la QoiS globale. La section 7 présente l'association des contraintes de dépendance au service intentionnel agrégat. Enfin, la section 8 conclut ce chapitre.

7.2 *VUE GLOBALE DE LA METHODOLOGIE DE PUBLICATION DE MiS-q*

La méthodologie de publication du modèle *MiS-q* que nous proposons s'inscrit dans la continuité de ce qui s'est fait dans l'approche iSOA (Kaabi,2007). Cette méthodologie se décline en plusieurs étapes :

Etape 1 : intégration de la qualité à la carte des besoins. Le modèle de la Carte (Rolland&Prakash,2000) est utilisé par Kaabi (Kaabi,2007) afin de capturer les exigences fonctionnelles des agents métier, et d'établir ainsi un couplage direct entre les buts métier et les services du système (*cf.* Chapitre 4). Afin de prendre en compte les exigences non-fonctionnelles des agents métier, nous proposons aux fournisseurs métier, à l'aide du

référentiel qualité (*cf.* Chapitre 5), d'associer des buts qualité à la carte des besoins, via des liens de contribution.

Etape 2 : génération de la QoiS simple. Les services intentionnels sont générés à partir de la carte des besoins (*cf.* Chapitre 4) (Kaabi,2007). Nous proposons aux fournisseurs métier de générer les QoiS simples, fournies par les services intentionnels atomiques, à partir de la carte des besoins et des liens de contribution. La QoiS est qualitative (avec des seuils de satisfaction) et elle exprime ce que les services intentionnels - services métier – permettent de fournir à l'agent métier.

Etape 3 : opérationnalisation des services atomiques. Un service intentionnel atomique peut être opérationnalisé par un ou plusieurs services logiciels *(i.e.* techniques d'implémentation) concurrents. Nous proposons aux fournisseurs métier d'identifier les services logiciels permettant d'aider l'utilisateur à la réalisation du but du service atomique, et ainsi obtenir leurs QoS. Le fournisseur métier sélectionne alors ceux qui l'intéressent et les traduits en services intentionnels atomiques. Après consultation du référentiel qualité, les QoSs, souvent quantitatives, sont aussi traduites en QoiS simple. Le service atomique initial peut alors devenir un service à choix alternatif.

Etape 4 : génération de la QoiS globale. Nous proposons aux fournisseurs métier un ensemble de règles méthodologiques lui permettant de générer la QoiS globale des services intentionnels agrégats. Le service agrégat ainsi obtenu doit être finalisé en identifiant les contraintes de dépendance existantes entre certaines variantes de ce service. Ces contraintes ont comme objectif de valider la cohérence des configurations possibles du service agrégat.

Etape 5 : l'association des contraintes de dépendance au service intentionnel. Dans la perspective de considérer la variabilité encapsulée dans le service intentionnel, nous proposons au fournisseur métier de définir des contraintes de dépendance (d'exigence et d'exclusion) entre les variantes des services. La définition de contraintes de dépendance permet de valider l'exécutabilité des différentes configurations du service agrégat (*cf.* Chapitre 5).

Une fois les différentes étapes appliquées, la description du service intentionnel est ainsi complète – elle comporte également ses contraintes de dépendance et ses qualités - et elle peut être publiée dans l'annuaire des services intentionnels. Nous proposons de développer ces différentes étapes dans ce qui suit.

7.3 *ETAPE 1 : INTEGRATION DE LA QUALITE A LA CARTE DES BESOINS*

La première étape de la méthodologie de publication du modèle *MiS-q* correspond à l'intégration de la qualité à la carte des besoins, afin de capter la qualité métier. Nous proposons au fournisseur métier de construire d'abord la carte des besoins, ensuite de consulter le référentiel qualité, afin de l'aider à identifier les buts qualité que la carte des besoins peut satisfaire partiellement. Enfin, intégrer les buts qualité à la carte des besoins. L'intégration de la qualité peut donc se faire comme suit :

1. Construire la carte des besoins
2. Consulter le référentiel qualité
3. Identifier les buts qualité
4. Intégrer les buts qualité à la carte des besoins

7.3.1 CONSTRUIRE LA CARTE DES BESOINS

Dans l'approche iSOA, Kaabi (Kaabi et *al.*,2004) (Kaabi,2007) a utilisé le modèle orienté but, et la Carte (Rolland&Prakash,2000) pour capturer les exigences fonctionnelles des agents métier. Selon Kaabi (Kaabi,2007), le modèle de la Carte est adapté à la découverte des services et de leur *variabilité métier*, car il met en évidence les différentes stratégies pour satisfaire le même but et aussi les différentes combinaisons de stratégies et de buts pour atteindre un but final. La description du modèle de la Carte, ainsi que la construction de la carte des besoins sont détaillées dans le Chapitre 4 et dans (Kaabi,2007).

7.3.2 CONSULTER LE REFERENTIEL QUALITE

Le référentiel qualité (*cf.* Chapitre 5) est un catalogue regroupant, par domaine, les buts qualité (tels que la sécurité, la performance et la précision), ainsi que les moyens permettant d'évaluer la satisfaction partielle de ces buts (les buts, les seuils de satisfaction, les métriques et les classes). Nous proposons que dans iSOA, les fournisseurs métier utilisent le référentiel qualité pour associer des qualités à la carte des besoins. Cette dépendance permet d'uniformiser, pour chaque domaine, les buts qualité devant être satisfaits partiellement, ainsi que les valeurs de satisfaction correspondantes.

Ainsi, le fournisseur métier souhaitant préciser les qualités auxquelles sa carte des besoins peut contribuer à satisfaire partiellement, doit consulter le référentiel qualité. En effet, le fournisseur localise d'abord son domaine d'activité (gestion de stock, automate d'analyse de sang, *etc*.). Ensuite, le fournisseur consulte les buts qualité pouvant être satisfaits

partiellement dans ce domaine, ainsi que les buts contribuant à la satisfaction partielle de ces buts qualité (*cf.* Chapitre 5 pour plus de détail). Ainsi, tous les fournisseurs métier d'un domaine ont accès à la même connaissance concernant les qualités que leurs services peuvent offrir, ainsi que les valeurs de satisfaction correspondantes. Ceci permet d'améliorer la crédibilité des fournisseurs.

7.3.3 IDENTIFIER LES BUTS QUALITE

Le fournisseur métier peut identifier les buts qualité que la carte des besoins peut satisfaire partiellement, de deux manières :

1. Le référentiel qualité précise les buts satisfaisant partiellement les buts qualité (*cf.* Chapitre 5). Dans ce cas, le fournisseur vérifie la similitude de ces buts (*i.e.* les buts simples du référentiel) avec les buts de la carte des besoins. S'il existe correspondance, le fournisseur peut considérer que les buts qualité satisfaits partiellement par ces buts peuvent être également satisfaits partiellement par ses buts ; ou
2. Le référentiel qualité ne précise pas les buts satisfaisant partiellement les buts qualité. Dans ce cas, le fournisseur définit, selon sa connaissance du métier, les qualités que sa carte des besoins peut satisfaire partiellement.

Le fournisseur métier doit choisir alors les *buts qualité simples* qui contribuent à satisfaire partiellement sa carte des besoins. Les buts qualité simples sont utilisés pour définir la qualité de service, car ils sont plus concis (plusieurs buts qualités simples pour un seul complexe), et diminuent le risque d'ambigüité. Ainsi, ils améliorent la sélection de services. Le fournisseur métier détermine également les seuils de satisfaction correspondant, en prenant comme référence ceux indiqués dans le référentiel qualité.

7.3.4 INTEGRER LA QUALITE A LA CARTE DES BESOINS

Nous considérons les divers niveaux d'abstraction de la carte des besoins, jusqu'à arriver aux sections opérationnalisables, qui reflètent l'opérationnalisation des buts qualité simple. L'intégration de la qualité à la carte des besoins correspond donc à la définition des liens de contribution reliant les sections opérationnalisables de la carte aux buts qualité simple. La Figure 88 illustre l'intégration des buts qualité, identifiés précédemment (*cf.* Section 7.3.3), à la carte des besoins (*cf.* Chapitre 4).

La partie haute de la Figure 88 regroupe l'ensemble des buts qualité que la carte permet de satisfaire partiellement. Ces buts qualité (élément But qualité dans le modèle *MiS-q*

(*cf.* Chapitre 6)) sont ceux identifiés dans la Section 3. Ils sont représentés sous forme d'un graphe de buts qualité, lorsqu'il s'agit de buts qualité complexes (*cf.* Chapitre 5). Nous précisons que ce graphe est un sous-graphe du référentiel, car il ne considère que les buts qualités (simples et complexes). La partie basse de la Figure 88 représente la carte des besoins (*cf.* Section 4.4), à partir de laquelle les services intentionnels (élément Service dans le modèle *MiS-q* (*cf.* Chapitre 6)) sont générés. La partie intermédiaire de la Figure 88 regroupe les liens de contributions (élément Lien de contribution dans le modèle *MiS-q* (*cf.* Chapitre 6)), qui sont représentés par des flèches allant des sections opérationnalisables de la carte vers les buts qualité simples du graphe des buts qualité. Ces liens expriment que chaque section opérationnalisable peut contribuer à satisfaire partiellement des buts qualité simples. Comme les buts qualité peuvent être satisfaits dans des limites acceptables, la sémantique des liens de contributions reflète cette satisfaction partielle. En effet, la section opérationnalisable contribue à satisfaire *(« + »)*, à très satisfaire *(« ++ »)*, à peu satisfaire *(« - »)* ou à très peu satisfaire *(« -- »)* les buts qualité (attribut Seuil de satisfaction dans le modèle *MiS-q* (*cf.* Chapitre 6)).

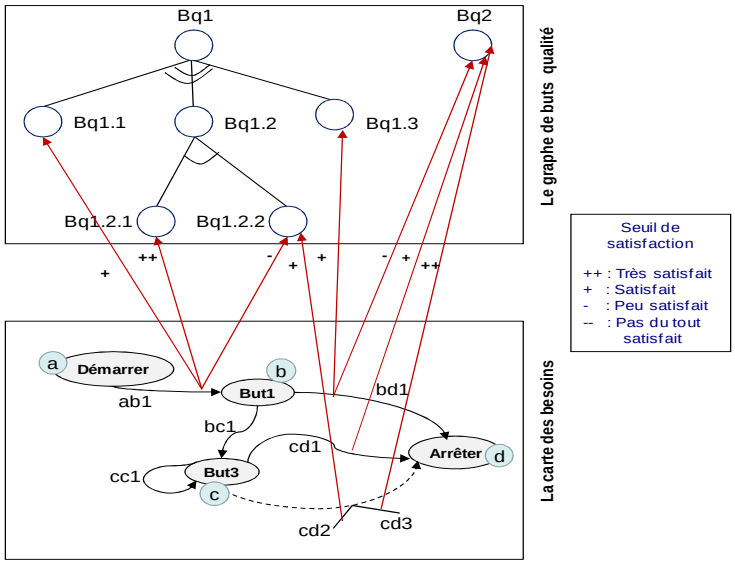

Figure 88. Exemple de représentation de l'intégration de la qualité à une carte de besoins

Afin de compléter la définition de ses services intentionnels, le fournisseur métier peut ainsi intégrer des buts qualité à la carte des besoins. Nous proposons à la Figure 89 un algorithme permettant d'aider le fournisseur métier à effectuer l'intégration des buts qualité à la carte des besoins. Le fournisseur de service détermine pour chaque section opérationnalisable de la carte, les liens de contribution pouvant exister avec les buts qualité simple. Il doit préciser la valeur de chaque lien de contribution, laquelle correspond au seuil de satisfaction « ++ », « + », « - » et « -- », fournie par la section opérationnalisable. Cette représentation permet

ainsi de faire ressortir la variabilité métier à travers les relations de paquet, de multi-segment et de multi-chemin. Cette variabilité métier permet de mettre en évidence les différentes manières de réaliser un but, et de satisfaire partiellement des qualités.

Si la section n'est pas opérationnalisable, il faut effectuer un affinement de la section (*cf.* Chapitre 4) et appliquer de manière récursive l'intégration des buts qualité (*i.e.* l'algorithme de la Figure 89) à la carte obtenue.

```
IntegrationQualité(C)
Entrée
C : la carte de besoins de niveau i. C est composée d'un ensemble de
sections, et de relations entre les sections (paquet,multi-segment,etc).
Sortie
ListeContribution : la liste des liens de contribution entre les
sections opérationnalisables et les buts qualité simple Qs

Début
Q : le graphe de buts qualité. Qs est l'ensemble des buts qualité
simples.
S : l'ensemble des seuils de satisfaction = {++, +,-, --}
L : représente le lien de contribution (section, Qs, S)
ListeContribution : ensemble de L
Pour chaque section ∈ C
 Si NOT opérationnalisable(C.section)
  Alors
  ListeContribution <- ListeContribution ∪ IntegrationQualité(C.section)
  Sinon
    Pour chaque Qs∈ Q
      Si ∃ L // existence de lien de contribution entre section et Qs
         Alors ListeContribution <- ListeContribution ∪ L
      Fin si
    Fin pour
 Fin si
Fin pour
```

Figure 89. Algorithme d'intégration de la qualité à la carte des besoins

Exemple

Afin d'illustrer l'intégration de la qualité dans la carte des besoins, nous reprenons l'exemple de l'approvisionnement de produits (*cf.* Chapitre 4).

La Figure 90 illustre l'intégration de la qualité à la carte réalisant le but *Effectuer l'approvisionnement en produits*. Nous proposons de considérer deux qualités principales, à savoir la Performance et la Fiabilité dans le système de l'approvisionnement de produits.

La carte des besoins de la Figure 90 permet de réaliser le but *Effectuer l'approvisionnement en produits*, tout en contribuant à la satisfaction partielle des buts qualité Performance et Fiabilité. Par exemple, la section cc_4 (*Contrôler le stock par Echantillonnage*), cc_5 (*Contrôler le stock par Périodique*) et cc_6 (*Contrôler le stock En continu*) contribuent respectivement « - », « + » et « ++ » à la satisfaction du but qualité Fiabilité. En effet, plusieurs sections (cc_4, cc_5 et cc_6) réalisant le même but (*i.e. Contrôler le stock*), mais avec des stratégies différentes,

auront différents seuils de satisfaction. La section cc_6 propose une stratégie de contrôle en continu. La section cc_5 propose une stratégie de contrôle périodique, tandis que la section cc_4 propose de prendre juste un échantillon. Chacune de ces sections permet de réaliser le même but (*i.e. Contrôler le stock*), mais elles contribuent différemment aux différentes qualités (la fiabilité dans ce cas) avec des seuils de satisfaction différents (« - », « + » et « ++»). Par conséquent, la sélection d'une section, parmi plusieurs, lesquelles réalisent le même but doit se baser sur les critères de qualité (*i.e.* les buts qualité). Des qualités sont alors intégrées à la carte des besoins, afin de prendre la dimension qualitative des services intentionnels.

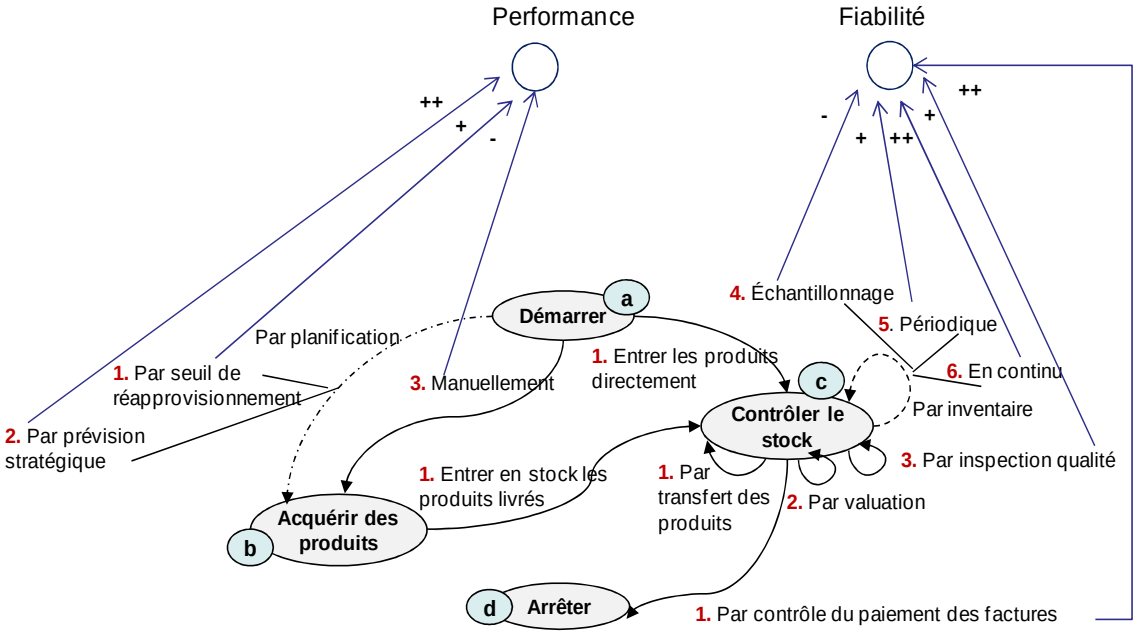

Figure 90. Intégration de la qualité à la carte *Effectuer l'approvisionnement en produits*

Une fois la carte des besoins construite, les contraintes et les qualités intégrées à la carte, nous proposons, dans ce qui suit, la génération des services intentionnels, leurs dépendances et leur qualité, tels que définis dans le modèle *MiS-q* (*cf.* Chapitre 6).

7.4 *ETAPE 2 : GENERATION DE LA QOIS SIMPLE*

La deuxième étape de la méthodologie de publication du modèle *MiS-q* consiste à proposer au fournisseur métier un ensemble de règles méthodologiques, afin de l'aider à générer les différents services intentionnels, ainsi que la QoiS simple. Cette étape consiste à :

1. générer les services intentionnels
2. générer la QoiS simple des services intentionnels atomiques

7.4.1 GENERER LES SERVICES INTENTIONNELS

Selon le modèle *MiS-q* (cf. Chapitre 6), le service intentionnel peut être atomique ou agrégat (composite ou à variation). Kaabi (Kaabi et *al.*,2004) (Kaabi,2007) a utilisé le modèle de la Carte (Rolland&Prakash,2000) pour modéliser les exigences fonctionnelles des utilisateurs. Le modèle de la Carte permet de combiner dans une unique représentation la perspective métier, correspondant aux buts et aux stratégies pour les atteindre, ainsi que la perspective système qui définit les services du système qui sont capables de supporter la réalisation de ces buts (Salinesi&Rolland,2003). Ainsi, Kaabi (Kaabi et *al.*,2004) (Kaabi,2007) propose d'associer d'une part à chaque section opérationnalisable de la carte un service atomique, et d'autre part à chaque relation entre les sections un type de service agrégat. La génération des services intentionnels est détaillée dans le Chapitre 4 et dans (Kaabi,2007).

7.4.2 GENERER LA QOIS SIMPLE

Nous avons présenté précédemment (*cf.* Section 7.3.4) que chaque section de la carte peut contribuer à la satisfaction partielle d'un ensemble de buts qualité. Dans le modèle *MiS-q* (*cf.* Chapitre 6), la qualité d'un service intentionnel (*QoiS-Quality of intentional Service*) est définie comme l'ensemble des buts qualité auxquels le service contribue à satisfaire partiellement. La QoiS est définie comme une QoiS simple, lorsque le service est atomique et comme une QoiS globale, lorsque le service est agrégat. Il s'agit, dans ce qui suit, de générer à partir de la carte des besoins et des liens de contribution la QoiS simple. La génération de la QoiS globale sera présentée à la Section 7.6.

Comme définie plus haut (*cf.* Chapitre 4), chaque section opérationnalisable de la carte correspond un service atomique. Egalement, dans le modèle *MiS-q*, la QoiS simple est associée à un service atomique (*cf.* Chapitre 6). Ainsi, nous proposons d'identifier la QoiS simple à partir de l'ensemble des liens de contribution reliant la section opérationnalisable aux buts qualité.

La Figure 91 met en évidence la représentation de la QoiS simple sur la carte des besoins.

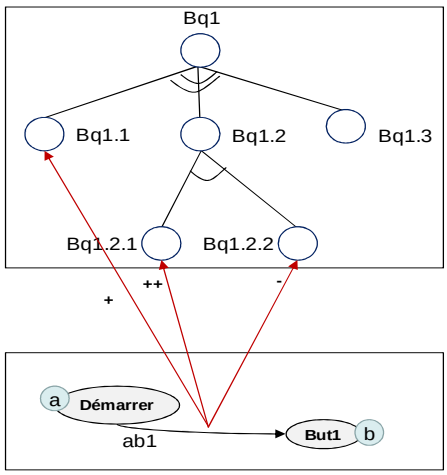

Figure 91. Représentation de la QoiS simple sur la carte des besoins

La documentation de la QoiS simple est formulée (Doc 1) dans le modèle *MiS-q* (*cf.* Chapitre 6) comme suit : $\boldsymbol{S}_a.\{< q_j, d_j >\}$. Par exemple, la section *ab1* (*cf.* Figure 91) contribue « + », « ++ » et «-» aux buts qualités *Bq1.1*, *Bq1.2.1* et *Bq1.2.2*. La QoiS simple correspond donc à : *ab1*. *{< Bq1.1, + >, < Bq1.2.1, ++ >, < Bq1.2.2, - >}*.

Cette définition est appliquée à l'exemple *Effectuer l'approvisionnement en produits* de la Figure 90 et nous avons identifié plusieurs QoiS simple, lesquelles sont récapitulées au Tableau 18. En effet, le Tableau 18 résume les services atomiques de l'exemple de l'approvisionnement de produits, tels que $\boldsymbol{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$ (correspondant à la section ab_1), $\boldsymbol{S}_{Etablir\ un\ inventaire\ en\ continu}$ (correspondant à la section cc_6) et $\boldsymbol{S}_{Contrôler\ le\ paiement\ des\ factures}$ (correspondant à la section cd_1), ainsi que les QoiS correspondantes, à savoir $\boldsymbol{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}.\{<Performance, +>\}$, $\boldsymbol{S}_{Etablir\ un\ inventaire\ en\ continu}.\{< Fiabilité, ++>\}$ et $\boldsymbol{S}_{Contrôler\ le\ paiement\ des\ factures}.\{< Fiabilité, +>\}$.

Tableau 18. Les QoiS simple de l'exemple *Effectuer l'approvisionnement en produits*

Section carte	Service atomique	QoiS simple	
ab1	$\boldsymbol{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}$	$QoiS(\boldsymbol{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement})$	$\boldsymbol{S}_{Commander\ des\ produits\ par\ seuil\ de\ réapprovisionnement}.$ {<Performance, +>}
ab2	$\boldsymbol{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}$	$QoiS(\boldsymbol{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques})$	$\boldsymbol{S}_{Commander\ des\ produits\ par\ prévisions\ stratégiques}.$ {<Performance, ++>}

ab3	$\mathbf{S}_{\text{Commander des produits manuellement}}$	QoiS($\mathbf{S}_{\text{Commander des produits manuellement}}$)	$\mathbf{S}_{\text{Commander des produits manuellement}}$. {<Performance,->}
cc3	$\mathbf{S}_{\text{Inspecter la qualité du stock}}$	QoiS($\mathbf{S}_{\text{Inspecter la qualité du stock}}$)	$\mathbf{S}_{\text{Inspecter la qualité du stock}}$. {<Fiabilité, +>}
cc4	$\mathbf{S}_{\text{Etablir un inventaire par échantillonnage}}$	QoiS($\mathbf{S}_{\text{Etablir un inventaire par échantillonnage}}$)	$\mathbf{S}_{\text{Etablir un inventaire par échantillonnage}}$. {<Fiabilité, ->}
cc5	$\mathbf{S}_{\text{Etablir un inventaire périodiquement}}$	QoiS($\mathbf{S}_{\text{Etablir un inventaire périodiquement}}$)	$\mathbf{S}_{\text{Etablir un inventaire périodiquement}}$. {<Fiabilité, +>}
cc6	$\mathbf{S}_{\text{Etablir un inventaire en continu}}$	QoiS($\mathbf{S}_{\text{Etablir un inventaire en continu}}$)	$\mathbf{S}_{\text{Etablir un inventaire en continu}}$. {<Fiabilité, ++>}
cd1	$\mathbf{S}_{\text{Contrôler le paiement des factures}}$	QoiS($\mathbf{S}_{\text{Contrôler le paiement des factures}}$)	$\mathbf{S}_{\text{Contrôler le paiement des factures}}$. {<Fiabilité, ++>}

7.5 *ETAPE 3 : OPERATIONNALISATION DES SERVICES ATOMIQUES*

La troisième étape de la méthodologie de publication du modèle *MiS-q* consiste à opérationnaliser le service intentionnel atomique. En effet, un service atomique donné peut être opérationnalisé par un ou plusieurs services logiciels (*i.e* techniques). Toutefois, comme défini dans le modèle *MiS-q* (*cf.* Chapitre 6), le service atomique est opérationnalisé par un seul service logiciel et fournit une seule QoiS simple. Pour décrire, dans les termes de *MiS-q*, les services logiciels, qui ne sont pas captés par le modèle de la Carte (celui-ci capte la variabilité métier), nous proposons au fournisseur métier de suivre les sous-étapes suivantes :

1. rechercher les services logiciels ;
2. sélectionner un (cas 1 à la Figure 92) ou plusieurs services logiciels (cas 2 à la Figure 92) ;
3. traduire les services logiciels en services intentionnels atomiques ; et
4. agréger les services atomiques, si nécessaire (*i.e.* cas 2), dans un service intentionnel à choix alternatif.

Ces différentes sous-étapes (rechercher, sélectionner, traduire et agréger) d'opérationnalisation des services atomiques sont résumées à la Figure 92, et seront détaillées dans ce qui suit. Ces sous-étapes sont réalisées pour chaque service intentionnel atomique abstrait.

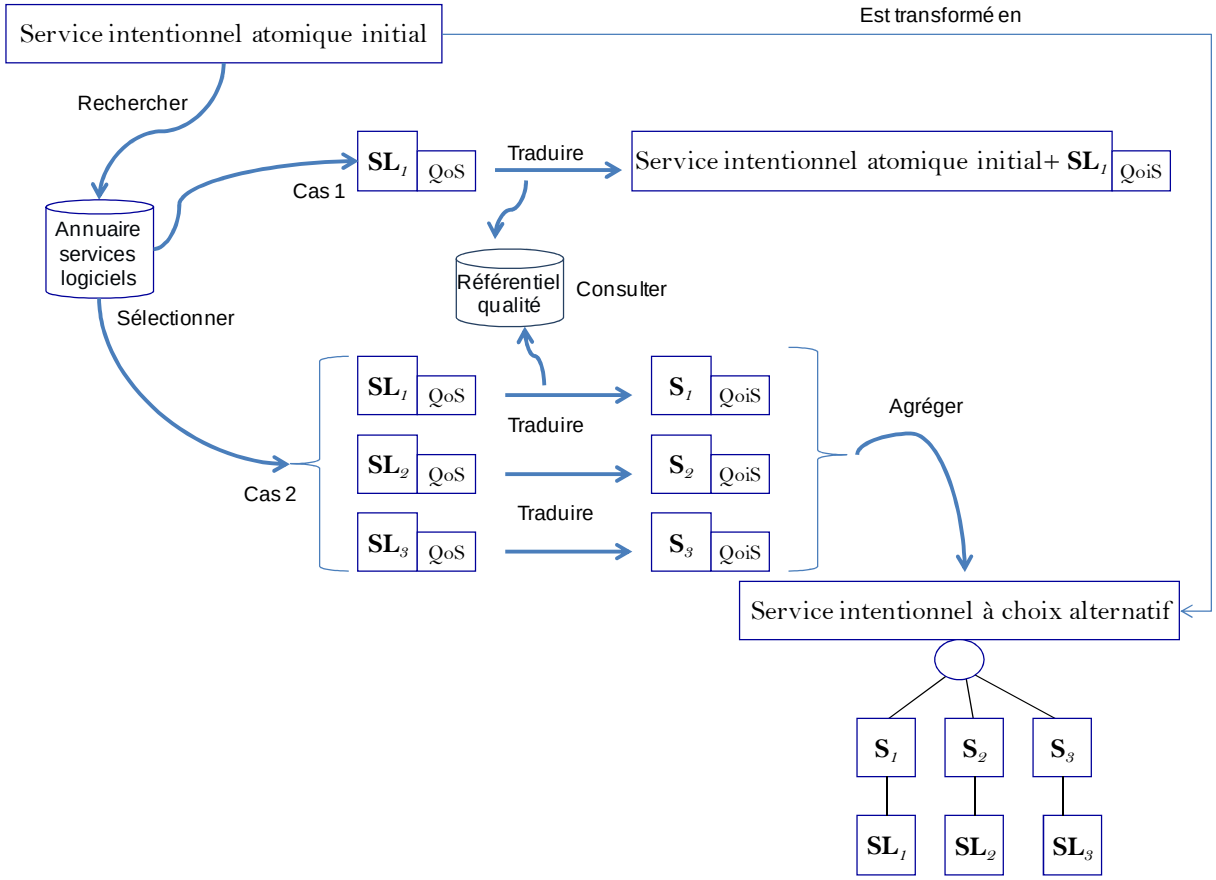

Figure 92. Les sous-étapes de l'opérationnalisation des services atomiques

7.5.1 RECHERCHER LES SERVICES LOGICIELS

Pour chaque service intentionnel atomique, le fournisseur métier recherche, dans l'annuaire, les services logiciels qui réalisent le but du service atomique. Cette recherche permet d'identifier les différentes techniques permettant d'opérationnaliser un service atomique donné. Plusieurs travaux proposent de rechercher des services Web par les buts (Guzélian,2007) (Mirbel&Crescenzo,2009)(Da Silva Santos et *al.*,2009). Dans l'architecture iSOA, Aljoumaa et ses collègues (Aljoumaa et *al.*, 2010) (Aljoumaa et *al.*, 2011) proposent une approche permettant de trouver les services logiciels qui réalisent le but du service atomique. La recherche dans l'annuaire permettrait potentiellement de retrouver plusieurs services logiciels, ainsi que leur QoS. Par exemple, le service atomique $\mathbf{S}_{\text{Authentifier un voyageur}}$ peut se faire en utilisant plusieurs techniques, telles que l'authentification par mot de passe, par une clé, ou par biométrie. Chacune de ces techniques fournit une QoS différente.

7.5.2 SELECTIONNER LES SERVICES LOGICIELS

Une fois les services logiciels qui réalisent le but du service atomique identifiés, le fournisseur métier sélectionne alors la ou les techniques (*i.e.* les services logiciels) qu'il souhaite considérer dans son système. Le choix des services logiciels à mettre en place dépend du domaine en question, du fournisseur qui va réaliser la technique, et des QoS fournies par les services logiciels. Par exemple, le fournisseur pourrait choisir l'utilisation de mot de passe et de la biométrie comme techniques d'authentification.

7.5.3 TRADUIRE LES SERVICES LOGICIELS

Selon, le modèle *MiS-q*, un service atomique est opérationnalisé par un seul service logiciel et offre une QoiS simple. Comme illustré à la Figure 92, deux cas peuvent se présenter :

Cas 1 : un seul service logiciel est choisi par le fournisseur. Dans ce cas, le service atomique initial garde son nom ou peut être augmenté du nom du service logiciel choisi. Par exemple, le fournisseur choisit d'utiliser la biométrie pour authentifier les voyageurs. Le service atomique initial (*i.e.* $\mathbf{S}_{\text{Authentifier un voyageur}}$) est alors augmenté de la technique d'authentification choisie, à savoir $\mathbf{S}_{\text{Authentifier un voyageur par biométrie}}$.

Cas 2 : plusieurs services logiciels sont choisis par le fournisseur. Dans ce cas, les différents services logiciels choisis sont traduits en des services atomiques, en les augmentant d'une description intentionnelle. Celle-ci consiste à rajouter la manière au but initial expliquant ainsi la technique utilisée pour la réalisation de ce but. Par exemple, l'utilisation de mot de passe comme technique d'authentification, devient le service atomique $\mathbf{S}_{\text{Authentifier un voyageur par mot de passe}}$. L'exécution du service atomique revient alors à l'exécution de ce service logiciel.

Dans les deux cas, la QoS fournie par le service logiciel choisi est traduite en une QoiS simple. Le fournisseur consulte le référentiel qualité (*cf.* Chapitre 5) de son domaine, afin d'exprimer la QoS du (ou des) service(s) logiciel(s) choisi(s), selon les termes du modèle *MiS-q* : en fonction des buts qualité simple (notés q_j) et des seuils de satisfaction (noté d_j), facilement compréhensibles par les agents métier. La QoS des services logiciels peut être quantitative (par exemple, temps de réponse égal à *60s*) ou qualitative (utilisation d'un protocole de cryptage, par exemple).

Par exemple, si l'authentification par mot de passe utilise l'algorithme de cryptage *BlowFish*[14], alors le seuil de satisfaction correspondant est « ++ » (après consultation du référentiel qualité). Ainsi le service atomique $\mathbf{S}_{\text{Authentifier un voyageur par mot de passe}}$ offre de la

[14] http://www.rd.cri74.org/repository/securite/algo_chiffrement.pdf

confidentialité avec le seuil « ++ », à savoir QoiS($\mathbf{S}_{\text{Authentifier un voyageur par mot de passe}}$)= $\mathbf{S}_{\text{Authentifier un voyageur par mot de passe}}$.{<Confidentialité,++>}. La QoiS du service atomique obtenue est alors la traduction de la qualité fournie par le service logiciel l'opérationnalisant. Cette traduction est faite grâce au référentiel qualité, par tous les fournisseurs métier du domaine, permettant ainsi une meilleure crédibilité des QoiS publiées. Cette sous-étape est réalisée pour chaque service logiciel choisi.

7.5.4 AGREGER LES SERVICES ATOMIQUES

Pour le cas 2, le service atomique initial devient alors un service à choix alternatif (⊗), lequel regroupe les différentes variantes permettant de réaliser le but de ce service. Les services atomiques (*i.e.* les services logiciels augmentés d'une description intentionnelle) obtenus précédemment correspondent aux différentes alternatives possibles de réalisation du but du service à choix alternatif. Par exemple, le service initial d'authentification d'un voyageur $\mathbf{S}_{\text{Authentifier un voyageur}}$ devient le service à choix alternatif suivant : $\mathbf{S}_{\text{Authentifier un voyageur}}$ = ⊗ ($\mathbf{S}_{\text{Authentifier par mot de passe}}$, $\mathbf{S}_{\text{Authentifier par biométrie}}$). Ce service à variation fournit alors une QoiS variante, telle que définie au Chapitre 6, à savoir la QoiS($\mathbf{S}_{\text{Authentifier un voyageur}}$)= ⊗ (QoiS($\mathbf{S}_{\text{Authentifier par mot de passe}}$), QoiS($\mathbf{S}_{\text{Authentifier par biométrie}}$)).

7.6 *ETAPE 4 : GENERATION DE LA QOIS GLOBALE*

La quatrième étape de la méthodologie de publication du modèle *MiS-q* comprend un ensemble de règles méthodologiques permettant au fournisseur métier de générer les QoiS globales fournies par les différents services agrégats. Nous proposons de générer, dans un premier temps, la QoiS globale. Cette étape consiste à :

1. générer la QoiS globale associée à chaque type de relation de la carte des besoins
2. générer la QoiS globale d'une section non opérationnalisable Génération de la QoiS globale

7.6.1 GENERER LA QOIS GLOBALE

Tel que défini plus haut (*cf.* Chapitre 4), chaque type de relation de la carte des besoins (*i.e.* paquet, multi-segment, chemin et multi-chemin) correspond à un type de service agrégat (*i.e.* service composite et service à variation). Egalement, dans le modèle *MiS-q*, la QoiS globale est associée au service agrégat (*cf.* Chapitre 6) et donc à la carte des besoins. Ainsi, nous proposons de montrer l'identification de la QoiS globale à partir des différents types de relation de la carte des besoins.

Étant donné que la carte des besoins, est représentée par un graphe (*cf.* Chapitre 4), la définition de la QoiS globale correspond donc au parcours des différentes QoiS des chemins de ce graphe (*i.e.* la carte des besoins). A l'instar de Rohleder (Rohleder,2009), nous proposons d'utiliser, l'algorithme de MacNaughton et Yamada (MacNaugthon&Yamada,1960) pour générer les QoiS globales fournies par les différents chemins de la carte.

En fait, l'algorithme de MacNaughton et Yamada a permis d'une part dans (Kaabi,2007) (*cf.* Chapitre 4) de générer tous les services agrégats dans une carte, et d'autres part dans (Rohleder,2009) d'identifier toutes les QoS dans cette carte. Nous proposons d'adopter cet algorithme pour démontrer que les différentes QoiS globales de la carte concernent respectivement aux différents types de QoiS du service agrégat de cette carte. Aussi, nous tenons à préciser qu'il ne s'agit pas d'exécuter l'algorithme pour la génération des services et ensuite celles des QoiS, mais l'exécuter une seule fois pour générer les différents types de service et à chacun de ces services correspond le même type de QoiS.

Nous présentons alors, dans ce qui suit, l'algorithme de MacNaughton et Yamada, son application pour la génération des différentes QoiS globales, et enfin son application sur un exemple.

7.6.1.1 *PRESENTATION DE L'ALGORITHME DE MACNAUGHTON ET YAMADA*

L'algorithme de MacNaughton et Yamada (MacNaugthon&Yamada,1960) a été initialement appliqué pour identifier tous les chemins possibles dans un automate fini. C'est un algorithme de parcours dans un graphe, lequel permet de générer tous les chemins entre un nœud initial et un nœud final. Il a été adapté par Kaabi (Kaabi,2007) sur la carte des besoins, afin de générer les différents services agrégats (*cf.* Chapitre 4).

Nous présentons d'abord l'algorithme générique et nous l'adapterons par la suite à la génération des QoiS globales. Soient les données suivantes :

- *s* et *t* représentent respectivement le nœud initial et le nœud final.
- *Q* représente l'ensemble des nœuds intermédiaires incluant *s* et *t*.
- *P* représente l'ensemble des nœuds intermédiaires entre *s* et *t* excluant ces derniers.

La formule initiale utilisée pour découvrir toutes les combinaisons possibles entre le nœud *s* et le nœud *t*, est la suivante :

$$Y_{s,Q,t} = \bullet(X^*_{s,\, Q\setminus\{s\},\, s}, X_{s,\, Q\setminus\{s,t\},\, t}, X^*_{t,\, Q\setminus\{s,t\},\, t}) \qquad (1)$$

Le symbole « ● » désigne l'opérateur de composition correspondant au séquencement des nœuds, tandis que le symbole « * » désigne l'opérateur d'itération, lequel précise si un nœud

est relié à (ou boucle sur) lui même. La formule $Y_{s,Q,t}$ indique l'ensemble des chemins possibles entre le nœud initial *s* et le nœud final *t.*

La formule *(1)* correspond à la composition du chemin itératif reliant le nœud *s* à lui même, en passant par l'ensemble *Q* sans passer par *s* (*i.e.* $X^*_{s,\ Q\setminus\{s\},\ s}$) ; composé au chemin allant du nœud *s* vers le nœud *t* , en passant par l'ensemble *Q* sans passer par *s* et *t* (*i.e.* $X_{s,\ Q\setminus\{s,t\},\ t}$) ; et composé au chemin itératif reliant le nœud *t* à lui même, en passant par l'ensemble *Q* sans passer par *s* et *t* (*i.e.* $X^*_{t,\ Q\setminus\{s,t\},\ t}$). Chaque *X* désigne un ensemble de chemins possibles d'un nœud vers un autre, tout en passant par un quelconque nœud intermédiaire. Soit un nœud particulier *q* de l'ensemble *P*, la formule générant l'ensemble des chemins possibles passant par un nœud intermédiaire *q* est la suivante :

$$X_{s,P,t} = \cup((X_{s,\ P\setminus\{q\},\ t}),\ \bullet(X_{s,\ P\setminus\{q\},\ q},\ X^*_{q,\ P\setminus\{q\},\ q},\ X_{q,\ P\setminus\{q\},\ t}) \qquad (2)$$

$$Si\ P=\emptyset\ alors\ X_{s,P,t} = X_{st}$$

La formule *(2)* définit l'ensemble des chemins entre les nœuds *s* et *t* en passant par le nœud intermédiaire *q*. Elle considère les chemins entre *s* et *t* sans passer par *q* (*i.e.* $X_{s,\ P\setminus\{q\},t}$), ainsi que les chemins qui passent par *q* (*i.e.* $\bullet(X_{s,\ P\setminus\{q\},\ q},\ X^*_{q,\ P\setminus\{q\},\ q},\ X_{q,\ P\setminus\{q\},\ t}$). Chaque X de la formule *(1)* est développé récursivement en utilisant la formule *(2)* et en prenant, au hasard, un but intermédiaire *q* de l'ensemble *P*. Le processus s'arrête lorsque l'ensemble *P* est vide. L'algorithme de MacNaughton et Yamada est résumé à la Figure 93.

```
Algorithme de MacNaughton et Yamada (Ci : carte)
Entrée
- s et t représentent le nœud initial et le nœud final.
- Q représente l'ensemble des buts incluant s et t.
- P représente l'ensemble des buts excluant s et t.
Sortie
Y_{s,Q,t} : représente tous les chemins, en se basant sur les formules de type
X_{st}.

Début
Y_{s,Q,t} = •(X*_{s, Q\{s}, s}, X_{s, Q\{s,t}, t}, X*_{t, Q\{s,t}, t})
Pour chaque X ∈ Y_{s,Q,t}
       Tant que (P ≠ Ø)
          Soit un nœud intermédiaire quelconque q ∈ P ;
          X_{s,P,t} = ∪ ((X_{s, P\{q}, t}), •(X_{s, P\{q}, q}, X*_{q, P\{q}, q}, X_{q, P\{q}, t}) ;
          // X_{s,P,t} est l'ensemble des chemins entre s et t en passant par q
          Si (pas de chemin de s vers t) alors X_{s,P,t} = Ø;
       Fin Tant que
       Si (P= Ø) Alors X_{s,P,t} = X_{st} ;
Fin
```

Figure 93. Algorithme de MacNaughton et Yamada

7.6.1.2 *ADAPTATION DE L'ALGORITHME DE MACNAUGHTON ET YAMADA A LA GENERATION DES QOIS GLOBALES*

La carte regroupe plusieurs QoiS globales relatives au nombre de chemins existants. Néanmoins, il est difficile d'identifier de manière exhaustive toutes les possibilités, en termes de QoiS. La connaissance des différentes QoiS globales nous renseigne sur la qualité des services agrégats, laquelle joue un rôle important dans la sélection d'un service plutôt qu'un autre.

L'algorithme de MacNaughton et Yamada a été utilisé pour identifier les différents chemins possibles dans une carte, reliant le but *Démarrer* au but *Arrêter*. Les différents chemins sont définis comme des services agrégats (*cf.* Chapitre 4). Nous proposons d'utiliser l'algorithme de MacNaughton et Yamada pour générer les différentes QoiS globales possibles, correspondant aux différents chemins de la carte des besoins, reliant le but *Démarrer*, noté *s*, au but *Arrêter*, noté *t*. Les QoiS de la carte sont décrites au départ en fonction des différentes formules mathématiques propres au formalisme de l'algorithme citées plus haut (*i.e.* les formules *(1)* et *(2)*). Par application de la formule initiale *(1)*, nous obtenons la formule *(1)'*:

$$QoiS(Y_{s,Q,t}) = \bullet(QoiS(X^*_{s,\, Q\setminus\{s\},\, s}),\ QoiS(X_{s,\, Q\setminus\{s,t\},\, t}),\ QoiS(X^*_{t,\, Q\setminus\{s,t\},\, t})) \qquad (1)'$$

La formule $QoiS(Y_{s,Q,t})$ correspond à toutes les combinaisons de QoiS possibles correspondant aux différents chemins allant du but *s* vers *t*, en passant par les buts *Q*. La $QoiS(Y_{s,Q,t})$ correspond à la composition de : la QoiS du chemin allant de *s* vers *s* en passant par l'ensemble des buts intermédiaires *Q* sans passer par le but *s* ; et à la QoiS du chemin allant de *s* vers *t* sans passer par *s* et *t* ; et la QoiS de tous les chemins de *t* vers *t* sans passer par *s* et *t.*

Chaque $QoiS(X)$ désigne la QoiS d'un ensemble de chemins possibles, reliant un but source à un but cible, en passant par un but intermédiaire quelconque. Soit un but *q* de l'ensemble *P* (*i.e.* les buts intermédiaires), la $QoiS(X_{s,P,t})$ est la formule générant l'ensemble des QoiS possibles correspondant aux chemins allant de *s* vers *t*, tout en passant par un but intermédiaire *q*. Par application de la formule *(2)*, nous obtenons la formule *(2)'* suivante:

$$QoiS(X_{s,P,t}) = \cup((QoiS(X_{s,\, P\setminus\{q\},\, t})), \bullet(QoiS(X_{s,\, P\setminus\{q\},\, q}), QoiS(X^*_{q,\, P\setminus\{q\},\, q}), QoiS(X_{q,P\setminus\{q\},\, t}))$$

$$\text{Si } P = \emptyset \text{ alors } QoiS(X_{s,P,t}) = QoiS(X_{s,t}) \qquad (2)'$$

La $QoiS(X_{s,P,t})$ représente la QoiS des chemins qui ne passent pas par *q* (*i.e.* $QoiS(X_{s,\, P\setminus\{q\},\, t})$) et la QoiS des chemins qui passent par *q* (*i.e.* $\bullet(QoiS(X_{s,\, P\setminus\{q\},\, q}), QoiS(X^*_{q,\, P\setminus\{q\},\, q}), QoiS$

($X_{q,P\setminus\{q\},t}$)). La formule *(2)'* est appliquée récursivement à chaque *QoiS* (X) de la formule *(1)'* jusqu'à ce que l'ensemble *P* soit vide.

7.6.1.3 *EXEMPLE*

Nous appliquons la formule *(1)'* sur l'exemple de la carte *Effectuer l'approvisionnement en produits* de la Figure 90, sachant que le but *Démarrer* est *a*, le but *Arrêter* est *d* et l'ensemble *Q* correspond à *{a, b, c, d}*. Nous obtenons la formule *(3)* suivante :

$$QoiS(Y_{a,\{a,b,c,d\},d}) = \bullet(QoiS(X^*_{a,\{b,c,d\},a}), QoiS(X_{a,\{b,c\},d}), QoiS(X^*_{d,\{b,c\},d})) \qquad (3)$$

La $QoiS(Y_{a,\{a,b,c,d\},d})$ correspond aux différentes combinaisons possibles de QoiS correspondant aux différents chemins allant de *a* (*i.e.* le but initial *Démarrer)* vers *d* (*i.e.* le but final *Arrêter*) en passant par l'ensemble *Q* des buts *{a, b, c, d}*. Nous appliquons la formule *(2)'* à chaque *QoiS(X)* de la formule *(3)* et nous obtenons les formules *(4)*, *(5)* et *(6)* du Tableau 19.

Tableau 19. Résultat de $QoiS(Y_{a,\{a,b,c,d\},d})$

Formule	
$QoiS(X_{a,\{b,c,d\},a}) = \cup(QoiS(X_{a,\{b,d\},a}), \bullet(QoiS(X_{a,\{b,d\},c}), QoiS(X^*_{c,\{b,d\},c}), QoiS(X_{c,\{b,d\},a})))$	(4)
$QoiS(X_{a,\{b,c\},d}) = \cup(QoiS(X_{a,\{b\},d}), \bullet(QoiS(X_{a,\{b\},c}), QoiS(X^*_{c,\{b\},c}), QoiS(X_{c,\{b\},d})))$	(5)
$QoiS(X_{d,\{b,c\},d}) = \cup(QoiS(X_{d,\{b\},d}), \bullet(QoiS(X_{d,\{b\},c}), QoiS(X^*_{c,\{b\},c}), QoiS(X_{c,\{b\},d})))$	(6)

En vérifiant la carte de la Figure 90, nous remarquons qu'il n'existe pas de chemins possibles partant de *a* vers *a* ou allant de *d* vers *d*. Par conséquent, il n'existe pas de QoiS et les formules *(4)* et *(6)* sont alors égales à l'ensemble vide :

$$QoiS\ (X_{a,\{b,c,d\},a}) = \emptyset \qquad (7)$$

$$QoiS\ (X_{d,\{b,c\},d}) = \emptyset \qquad (8)$$

Par contre, il existe plusieurs chemins possibles entre *a* et *d* (en passant par *{b,c}*}, donc il existe aussi plusieurs QoiS possibles relatives à ces chemins. à partir des formules *(3)*, *(5)*, *(7)* et *(8)* nous déduisons l'équation *(9)* :

$$QoiS\ (Y_{a,\{a,b,c,d\},d}) = QoiS\ (X_{a,\{b,c\},d}) \qquad (9)$$

Nous appliquons la formule *(2)'* à chaque *QoiS(X)* de la *QoiS* ($X_{a,\{b,c\},d}$) de la formule *(5)* et nous obtenons les formules du Tableau 20.

Tableau 20. Résultat de *QoiS* ($X_{a,\{b,c\},d}$)

Formule initiale	**Formule finale**
QoiS ($X_{a,\{b\},d}$)= ∪ (QoiS (X_{ad}), ● (QoiS (X_{ab}), QoiS (X^*_{bb}), QoiS (X_{bd})))	QoiS ($X_{a,\{b\},d}$)= Ø
QoiS ($X_{a,\{b\},c}$) = ∪ (QoiS (X_{ac}), ● (QoiS (X_{ab}), QoiS (X^*_{bb}), QoiS (X_{bc})))	QoiS ($X_{a,\{b\},c}$) = ∪ (QoiS (X_{ac}), ●(QoiS (X_{ab}), QoiS (X_{bc})))
QoiS ($X^*_{c,\{b\},c}$) = ∪(QoiS (X_{cc}), ●(QoiS (X_{cb}), QoiS (X^*_{bb}), QoiS (X_{bc})))	QoiS ($X_{c,\{b\},c}$) = QoiS (X_{cc})
QoiS ($X_{c,\{b\},d}$) = ∪ (QoiS (X_{cd}), ● (QoiS (X_{cb}), QoiS (X^*_{bb}), QoiS (X_{bd})))	QoiS ($X_{c,\{b\},d}$) = QoiS (X_{cd})

À partir de la formule *(9)* et des différentes formules du Tableau 20, nous obtenons la formule *(10)* qui est égale à :

$$QoiS\ (Y_{a,\{a,b,c,d\},d}) = \bullet(\cup\ (QoiS\ (X_{ac}),\ \bullet\ (QoiS\ (X_{ab}),\ QoiS\ (X_{bc}))),\ QoiS(X^*_{cc}),\ QoiS\ (X_{cd}))) \qquad (10)$$

La formule *(10)* indique de façon générale que la QoiS de la carte des besoins permettant de partir du but *Démarrer (*noté *a)* au but *Arrêter (*noté *d)* correspond à la QoiS d'une séquence de chemins. Elle correspond particulièrement :

- à la QoiS du chemin qui passe par le but *Acquérir des produits* (noté *b*). Dans ce cas, la $QoiS\ (Y_{a,\{a,b,c,d\},d}) = \bullet(QoiS\ (X_{ab}),\ QoiS\ (X_{bc}),\ QoiS(X^*_{cc}),\ QoiS\ (X_{cd}))$; ou
- à la QoiS du chemin qui passe par le but *Contrôler le stock* (noté *c*). Dans ce cas, la $QoiS\ (Y_{a,\{a,b,c,d\},d}) = \bullet(QoiS\ (X_{ac}),\ QoiS(X^*_{cc}),\ QoiS\ (X_{cd}))$.

En vérifiant la carte de la Figure 90, nous remarquons qu'il n'existe pas de QoiS relatives aux chemins partant du but *Démarrer* (noté *a*) vers le but *Contrôler le stock* (noté *c*) ou du but *Acquérir des produits* (noté *b*) vers le but *Contrôler le stock* (noté *c*). Autrement dit, il n'existe pas de liens de contribution des sections *ac1* et *bc1* à la satisfaction partielle des buts qualité *Rapidité* et *Fiabilité*. Donc :

$$QoiS\ (X_{ac}) = Ø \quad \text{et} \quad QoiS\ (X_{bc}) = Ø \qquad (11)$$

Par conséquent, *(10)* et *(11)* donnent :

$$QoiS\ (Y_{a,\{a,b,c,d\},d}) = \bullet(QoiS\ (X_{ab}), QoiS(X^*_{cc}), QoiS\ (X_{cd})) \qquad (12)$$

La formule *(12)* indique, par rapport aux buts qualités définis dans la carte de la Figure 90, que la QoiS de la carte du service $\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}$ correspond à la QoiS du chemin allant de *a* vers *b* (*i.e. QoiS* (X_{ab})), ajoutée à la QoiS du chemin allant de *c* vers *c* (*i.e. QoiS* (X^*_{cc})), et augmentée à la QoiS du chemin allant de *c* vers *d* (*i.e. QoiS* (X_{cd})).

Afin de définir la QoiS globale selon les termes du modèle *MiS-q*, nous proposons d'abord d'effectuer la correspondance entre les *QoiS*(X_{st}) et la carte des besoins. Ensuite, nous définissons les différents types de QoiS globales.

7.6.1.4 *CORRESPONDANCE ENTRE LES FORMULES QOIS(X_{ST}) ET LES RELATIONS DE LA CARTE*

Les formules de type X_{st} représentent des sections et des relations de la carte, à savoir le paquet, le multi-segment, le chemin et le multi-chemin (*cf.* Chapitre 4). Par conséquent, nous proposons que les formules de type *QoiS*(X_{st}) correspondent aux QoiS des sections et aux QoiS des relations de la carte, à savoir la QoiS d'un paquet, la QoiS d'un multi-segment, la QoiS d'un chemin et la QoiS d'un multi-chemin. La Figure 94 présente l'algorithme permettant d'effectuer une mise en correspondance entre les formules *QoiS*(X_{st}) et les relations de la carte.

Algorithme de correspondance (*QoiS* (X_{st}))
Entrée
QoiS(X_{st}) correspond à la qualité de la carte
Sortie
QoiS des relations de la carte, à savoir QoiS de paquet, QoiS de multi-segment, QoiS de chemin et QoiS de multi-chemin

Début
Si X_{st} correspond à une section opérationnalisable,
Alors la *QoiS(X_{st}) est une* QoiS simple;

Si X_{st} correspond à un paquet (P) ou un multi-segment (MS)
Alors la *QoiS(X_{st}) est respectivement une* QoiS de paquet (QoiS(P)) ou une QoiS de multi-segment (QoiS(MS));

Si X_{st} correspond à un chemin (C) relié par le symbole de composition ''•''
Alors la *QoiS(X_{st}) est* une QoiS de chemin (QoiS(C));

Si X_{st} correspond à un multi-chemin (MC) relié par le symbole de variation ''∪''
Alors la *QoiS(X_{st}) est* une QoiS multi-chemin (QoiS(MC));

Si X_{st} est itérative
Alors la *QoiS(X_{st}) est* une QoiS itérative et optionnelle (QoiS(X^*_{st})).
Fin

Figure 94. Algorithme de mise en correspondance entre *QoiS*(X_{st}) et la carte

Par application de l'algorithme présenté ci-dessus (*cf.* Figure 94) sur la carte *Effectuer l'approvisionnement en produits* de la Figure 90, nous obtenons les résultats du Tableau 21.

Tableau 21. Liste de la QoiS des types de relation de la carte *Effectuer l'approvisionnement en produits*

Relation de la carte	**QoiS de relation**	**Exemple de QoiS de relation**
Paquet (*P*)	QoiS de Paquet (*QoiS(P)*)	$QoiS(P_{ab})$ $QoiS(P_{cc})$
Multi-Segment (*MS*)	QoiS de Multi-Segment (*QoiS(MS)*)	$QoiS(MS_{ab})$ $QoiS(MS_{cc})$
Chemin (*C*)	QoiS de Chemin (*QoiS(C)*)	$QoiS(C_{a,\{b\},c})$ $QoiS(C_{a,\{b,c\},d})$
Multi-Chemin (*MC*)	QoiS de Multi-Chemin (*QoiS(MC)*)	$QoiS(MC_{a,\{b\},c})$

Selon le Tableau 21, par exemple, la QoiS($C_{a,\{b\},c}$) correspond à la qualité du chemin $C_{a,\{b\},c}$. Sachant que $C_{a,\{b\},c}$ = ●(MS_{ab}, bc_1), alors la QoiS($C_{a,\{b\},c}$) est égale à la QoiS(●(MS_{ab}, bc_1)).

Ainsi, la formule (*12*) est égale à :

$$QoiS\ (\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}) = \bullet(QoiS\ (MS_{ab}),\ QoiS(MS^*_{cc}),\ QoiS\ (cd1))$$

7.6.1.5 *IDENTIFICATION DE LA QOIS GLOBALE*

Les relations de la carte, à savoir le paquet (*P*), le multi-segment (*MS*), le chemin (*C*) et le multi-chemin (M*C*) représentent les différents types de services agrégats, à savoir le service à variation à choix alternatif (*i.e.* le paquet), le service à variation à choix multiple (*i.e.* le multi-segment), le service composite séquentiel (*i.e.* le chemin), le service à variation de chemin (*i.e.* le multi-chemin) (*cf.* Chapitre 4).

Nous proposons d'identifier les différents types de QoiS globale, définis dans le modèle *MiS-q* (*i.e.* QoiS variante à choix alternatif, QoiS variante à choix multiple, QoiS variante de chemin, QoiS composite séquentielle) (*cf.* Chapitre 6), à partir des QoiS(*P*), QoiS(*MS*), QoiS(*C*) et QoiS(*MC*), présentées précédemment (*cf.* Section 7.6.1.4).

- La QoiS variante à choix alternatif est une QoiS de paquet correspondant à la QoiS du service à variation à choix alternatif ;
- La QoiS variante à choix multiple est une QoiS de multi-segment correspondant à la QoiS du service à variation à choix multiple ;

- La QoiS composite séquentielle est une QoiS de chemin correspondant à la QoiS du service composite séquentiel ; et
- La QoiS variante de chemin est une QoiS de multi-chemin correspondant à la QoiS du service à variation de chemin.

Nous reprenons l'exemple de la carte *Effectuer l'approvisionnement en produits* (*cf.* Figure 90) et nous en déduisons les types de QoiS globale qui sont résumés au Tableau 22.

Tableau 22. Liste des types de QoiS globale de la carte *Effectuer l'approvisionnement en produits*

QoiS de type de relation	**QoiS de service**	**Type de QoiS globale**
$QoiS(P_{ab})$ $QoiS(P_{cc})$	$QoiS(\mathbf{S}_{\text{Acquérir des produits par planification}})$ $QoiS(\mathbf{S}_{\text{Etablir un inventaire physique}})$	QoiS variante à choix alternatif
$QoiS(MS_{ab})$ $QoiS(MS_{cc})$	$QoiS(\mathbf{S}_{\text{Acquérir des produits}})$ $QoiS(\mathbf{S}_{\text{Contrôler le stock}})$	QoiS variante à choix multiple
$QoiS(C_{a,\{b\},c})$ $QoiS(C_{a,\{b,c\},d})$	$QoiS(\mathbf{S}_{\text{Entrer les produits en stock normalement}})$ $QoiS(\mathbf{S}_{\text{Effectuer l'approvisionnement en produits}})$	QoiS composite séquentielle
$QoiS(MC_{a,\{b\},c})$	$QoiS(\mathbf{S}_{\text{Recevoir le stock}})$	QoiS variante de chemin

Selon le Tableau 22, la formule (*12*) est égale à :

$$QoiS(\mathbf{S}_{Effectuer\ l'approvisionnement\ en\ produits}) = \bullet\ (v\ (QoiS(\mathbf{S}_{Pab})), QoiS(\mathbf{S}_{ab3}))), v\ (QoiS(\mathbf{S}_{cc1})^{*}, QoiS(\mathbf{S}_{cc2})^{*}, QoiS(\mathbf{S}_{cc3})^{*}, QoiS(\mathbf{S}_{Pcc})), QoiS(\mathbf{S}_{cd1}))$$

Nous pouvons déduire qu'une fois le service agrégat généré, à partir de la carte des besoins, en utilisant l'algorithme de MacNaughton et Yamada, il s'agit alors d'associer à chaque type de service une QoiS de même type, sans réutiliser cet algorithme. Par exemple, pour un service à choix multiple correspond une QoiS variante à choix multiple.

7.6.2 GENERER LA QOIS GLOBALE D'UNE SECTION NON OPERATIONNALISABLE

La QoiS d'une section non opérationnalisable correspond à la QoiS de la carte qui l'affine. La relation d'affinement (*cf.* Chapitre 4) dans une carte permet de décrire une section non opérationnalisable, à un niveau d'abstraction donné (*i*), par une carte à un niveau d'abstraction moins élevé (*i+1*) (*i.e.* une carte d'un niveau d'affinement plus élevé).

Nous proposons de suivre le même raisonnement au niveau (*i+1*) que celui effectué au niveau (*i*) (*cf.* Chapitre 4) pour l'identification de la QoiS simple et de la QoiS globale de la nouvelle carte. Le résultat obtenu au niveau (*i+1*) est transféré au niveau (*i*), suivant un raisonnement Botton-up.

Au départ, nous avons supposé que la *QoiS* $(X_{bc})=\emptyset$ (formule *(11)*). Par exemple, supposons maintenant que la section *<Acquérir des produits, Contrôler le stock, Entrer en stock des produits livrés >* (ayant le code bc_1), de la carte *Effectuer l'approvisionnement en produits* (*cf.* Figure 90), n'est pas une section opérationnalisable. Elle est alors affinée par la carte de la Figure 95.

En appliquant l'algorithme de MacNaughton et Yamada (*cf.* Section 7.6.1.2), nous en déduisons la QoiS globale de la carte du niveau d'abstraction (*i+1*). Celle-ci est égale à :

$$QoiS\ (C_{bc1})= \bullet\ (QoiS\ (\otimes(C_{bc1}.ab_1, C_{bc1}.ab_2, C_{bc1}.ab_3, C_{bc1}.ab_4)), \cup(QoiS\ (C_{bc1}.bd_1), QoiS\ (C_{bc1}.C_{b,\{c\},d})))$$

Nous proposons que la QoiS d'une section non opérationnalisable est égale à la QoiS de la carte qui l'affine. Ainsi, la QoiS de la section *<Acquérir des produits, Contrôler le stock, Entrer en stock des produits livrés >* du niveau d'abstraction (*i*) est égale à :

$$QoiS\ (C.\ C_{bc1})= \bullet\ (QoiS\ (\otimes(C_{bc1}.ab_1, C_{bc1}.ab_2, C_{bc1}.ab_3, C_{bc1}.ab_4)), \cup(QoiS\ (C_{bc1}.bd_1), QoiS\ (C_{bc1}.C_{b,\{c\},d})))$$

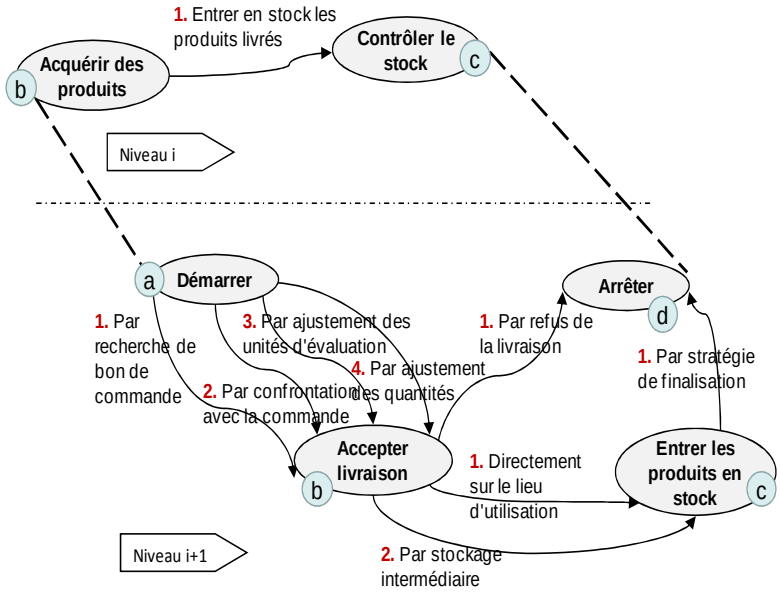

Figure 95. Affinement de la section *<Acquérir des produits, Contrôler le stock, Entrer en stock des produits livrés >*

Ainsi, nous en déduisons la nouvelles QoiS de la carte du niveau d'abstraction (*i*) (l'ancienne est celle de la formule (10)). Celle-ci prend en compte la QoiS de la section affinée. Elle est donc égale à :

$$QoiS\ (Y_{a,\{a,b,c,d\},d}) = \bullet(QoiS\ (X_{ab}),\ \bullet(QoiS\ (\otimes(C_{bc1}.ab1,\ C_{bc1}.ab2,\ C_{bc1}.ab3,\ C_{bc1}.ab4)),$$
$$\cup(QoiS\ (C_{bc1}.bd1),\ QoiS\ (C_{bc1}.Cb,\{c\},d))),\ QoiS(X^*_{cc}),\ QoiS\ (X_{cd})))$$

7.7 *ETAPE 5 : ASSOCIATION DES CONTRAINTES DE DEPENDANCE AU SERVICE INTENTIONNEL AGREGAT*

La cinquième étape de la méthodologie de publication du modèle *MiS-q* consiste à considérer la variabilité du service intentionnel agrégat, en définissant des contraintes de dépendance.

Le service intentionnel agrégat, via le service à variation, introduit de la variabilité dans la manière d'atteindre le but du service, afin de rendre explicite les variations possibles. Néanmoins, cette variabilité implique que toutes les combinaisons possibles ne sont pas forcément faisables ou acceptables. Pour cela, comme présenté dans le modèle *MiS-q* (*cf.* Chapitre 6), nous associons au service intentionnel agrégat en ensemble de contraintes de dépendance. Un service (atomique ou agrégat) peut alors influencer d'autres services (atomiques ou agrégats), via les contraintes de dépendance. Ces dernières sont définies entre les variantes du service agrégat de façon à garder, lors du choix de ces variantes, celles qui sont compatibles et exclure celles qui ne le sont pas. Ces contraintes, dans le contexte d'un service agrégat, peuvent être des contraintes qui expriment que le choix d'un service peut exiger le choix d'autres services (contrainte de type Exige), ou au contraire la sélection d'un service peut exclure la sélection d'autres services (contrainte de type Exclut). Elles permettent de valider l'exécutabilité des configurations du service agrégat (*cf.* Chapitre 6) et donc sa cohérence.

La contrainte de dépendance Exige exprime que la sélection d'un service implique la sélection d'un autre service. La contrainte de dépendance Exclut, quant à elle, définit que la sélection d'un service exclut la sélection d'un autre. Les contraintes de dépendance peuvent concerner aussi bien un service atomique qu'un service agrégat.

Afin d'associer des contraintes de dépendance aux services intentionnels, nous proposons au fournisseur métier d'effectuer ce qui suit :

1. identifier les contraintes de dépendance
2. décrire les contraintes de dépendance :
3. vérifier les règles de cohérence des contraintes de dépendance

7.7.1 IDENTIFIER LES CONTRAINTES DE DEPENDANCE

L'identification des contraintes de dépendance se fait en étudiant le système en question. Il peut exister des contraintes métier associées aux fonctionnalités du système. Aussi, il peut exister des contraintes techniques associées aux services logiciels. Par exemple, la mise en place d'une authentification par biométrie exige l'utilisation d'un lecteur d'empreintes. En reprenant l'exemple de système de réservation d'un billet de train, présenté plus haut (*cf.* dans le Chapitre 6), nous avons identifié plusieurs contraintes métier. En effet, la réservation d'un billet à prix réduit exige la présentation d'un justificatif de réduction, d'où la contrainte d'inclusion suivante : $\mathbf{S}_{\text{Réserver un billet de train réduit}}$ Exige $\mathbf{S}_{\text{Présenter une carte de réduction}}$. Aussi, les voyageurs possédant un abonnement Fréquence et souhaitant bénéficier d'une réduction du prix du billet, doivent effectuer leur réservation durant la période Bleu, d'où la contrainte d'inclusion suivante : ($\mathbf{S}_{\text{Posséder abonnement Fréquence}}$, $\mathbf{S}_{\text{Bénéficier de prix réduit}}$) Exige $\mathbf{S}_{\text{Choisir période Bleu}}$. Dans le cas de réservation d'un billet de train par Internet, le paiement du billet en espèce est exclu, d'où la contrainte d'exclusion suivante : $\mathbf{S}_{\text{Réserverr un billet de train par internet}}$ Exclut $\mathbf{S}_{\text{Payer un billet de train par espèce}}$. Enfin, les voyageurs possédant un abonnement Fréquence ne peuvent bénéficier d'une réduction du prix du billet, lorsqu'ils réservent durant la période Rouge, d'où la contrainte d'exclusion suivante : ($\mathbf{S}_{\text{Posséder abonnement Fréquence}}$, $\mathbf{S}_{\text{Choisir période Rouge}}$) Exclut $\mathbf{S}_{\text{Bénéficier de prix réduit}}$.

7.7.2 DECRIRE LES CONTRAINTES DE DEPENDANCE

La description des contraintes consiste à assimiler les contraintes de dépendance Exige et Exclut à la logique booléenne, telle que définie dans *MiS-q* (*cf.* Chapitre 6). Dans l'exemple de réservation de billet de train, nous proposons de décrire les contraintes identifiées comme suit : le voyageur ayant bénéficié d'une réduction lors de l'achat de son billet de train, doit systématiquement présenter un justificatif de réduction. Cette contrainte est décrite comme suit : $\mathbf{S}_{\text{Acheter un billet de train réduit}} \Rightarrow (\mathbf{S}_{\text{Présenter une carte de réduction}})$. Les voyageurs ayant un abonnement Fréquence et souhaitant bénéficier d'une réduction du prix du billet, doivent faire leur réservation durant la période Bleu. Cette contrainte est décrite comme suit : $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Bénéficier de prix réduit}} \Rightarrow (\mathbf{S}_{\text{Choisir période Bleu}})$. Le paiement d'un billet de train en espèce est interdit lorsque les voyageurs ont réservé leurs billets de train sur internet. La contrainte de dépendance correspondant à cet exemple est alors définie comme suit : $\mathbf{S}_{\text{Réserver un billet de train par internet}} \Rightarrow \neg(\mathbf{S}_{\text{Payer un billet de train par espèce}})$. Enfin, les voyageurs ayant un abonnement Fréquence ne peuvent bénéficier d'une réduction du prix du billet, lorsqu'ils réservent durant la période

Rouge. Cette contrainte est décrite comme suit : $\mathbf{S}_{\text{Posséder abonnement Fréquence}} \wedge \mathbf{S}_{\text{Choisir période Rouge}} \Rightarrow \neg (\mathbf{S}_{\text{Bénéficier de prix réduit}})$.

7.7.3 VERIFIER LES REGLES DE COHERENCE DES CONTRAINTES DE DEPENDANCE

Une fois les contraintes de dépendance décrites, il s'agit alors de valider la description de ces contraintes, en vérifiant les règles de cohérence. Celles-ci consistent, d'abord, à vérifier que les contraintes comprennent une seule cible. Si plusieurs cibles sont introduites alors réécrire les contraintes comme décrit dans le Chapitre 6. Ensuite, vérifier que les contraintes respectent la sémantique des opérateurs d'agrégation. Par exemple, une contrainte d'exigence ou d'exclusion ne doit pas exister entre les variantes d'un service à choix alternatif (*cf.* Chapitre 6). De plus, l'assimilation des contraintes à la logique booléenne permet de valider la cohérence des contraintes.

7.8 *CONCLUSION*

Nous avons présenté dans ce chapitre la méthodologie de publication du modèle *MiS-q*, qui s'inscrit dans la continuité de ce qui s'est fait dans l'approche iSOA (Kaabi,2007). En effet, le modèle de la Carte est utilisé par Kaabi (Kaabi,2007) pour capturer les exigences fonctionnelles des agents métier, et d'établir ainsi un couplage direct entre les buts métier et les services du système. (*cf.* Chapitre 4).

Dans la perspective de capturer les exigences non-fonctionnelles des agents métier, nous avons proposé d'intégrer la qualité à la carte des besoins, en lui intégrant des buts qualité, via des liens de contribution.

A l'instar de la génération des services intentionnels (Kaabi,2007), nous avons proposé des règles méthodologiques permettant, à partir des liens de contribution, de générer la QoiS des services intentionnels. Aussi, nous avons proposé d'opérationnaliser chaque service atomique.

Enfin, afin de prendre en considération la variabilité du service intentionnel, nous avons proposé d'associer aux variantes des services, des contraintes de dépendance. Celles-ci expriment que le choix d'un service peut exiger ou exclure le choix d'un autre service. Ces contraintes permettent de valider l'exécutabilité des configurations du service agrégat.

Une fois, la description du service intentionnel complète, elle peut alors être publiée dans l'annuaire des services intentionnels.

CHAPITRE 8 : SELECTION INTENTIONNELLE DES SERVICES

8.1 *INTRODUCTION*

Nous avons présenté, dans le Chapitre 6, le modèle intentionnel de la qualité du service, à savoir le *MiS-q*. Grâce à ce modèle, le service intentionnel peut, en plus du but qu'il réalise, offrir une qualité de service (*QoiS-Quality of intentional Service*) correspondant à l'ensemble des buts qualité que le service contribue à satisfaire partiellement. Le service intentionnel atomique offre une QoiS simple, tandis que le service agrégat fournit une QoiS globale. La QoiS est mise à la disposition de l'agent métier grâce à sa publication dans l'annuaire des services intentionnel. Pour le service agrégat, une $QoiS_{min}$ est calculée, laquelle est utilisée dans la sélection des services. Une $QoiS_{moy}$ est également calculée pour comparer les configurations d'un service agrégat.

A partir de ces informations, ce chapitre consiste à proposer une méthodologie de sélection et de configuration intentionnelle, permettant de guider l'exécution intentionnelle des services, de façon à choisir les services les mieux adaptés, en fonction des exigences fonctionnelles et non-fonctionnelles des clients. Celles-ci sont modélisées sous forme d'un contexte qualité, afin de faciliter le processus de sélection. Le processus de sélection intentionnelle permet de classer un ensemble de services, appartenant à différents fournisseurs, en fonction des préférences de l'agent métier. Si le résultat comporte de la variabilité, nous proposons de configurer les services agrégats. Le processus de configuration intentionnelle permet d'abord de générer les configurations exécutables du service agrégat, en tenant compte des contraintes de dépendance associées aux services. Ce processus permet ensuite de classer ces différentes configurations, de façon à se rapprocher davantage du contexte qualité de l'agent métier.

Pour cela, nous proposons d'adapter la méthode multicritère d'aide à la décision TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) afin d'effectuer la sélection et la configuration intentionnelle des services.

Ce chapitre est composé de plusieurs sections. La section 2 présente la vue globale de la méthodologie de sélection intentionnelle des services. La section 3 met en évidence le modèle

du contexte qualité du client. La section 4 définit le processus de sélection intentionnelle des services, tandis que la section 5 présente le processus de configuration intentionnelle du service agrégat. La section 6 discute les méthodes d'aide à la décision multicritères et la section 7 décrit la méthode TOPSIS, ainsi que son adaptation à la sélection et à la configuration intentionnelle des services. Enfin, la section 8 conclue ce chapitre.

8.2 *VUE GLOBALE DE LA METHODOLOGIE DE SELECTION INTENTIONNELLE DES SERVICES*

La Figure 96 présente une vue globale de la méthodologie de sélection intentionnelle des services. Celle-ci consiste à choisir les services intentionnels correspondant au mieux au contexte qualité du client.

Le service intentionnel, tel que défini plus haut (*cf.* Chapitre 6), est exprimé de façon intentionnelle, en mettant en avant les buts et les stratégies que l'entreprise choisit pour les atteindre. Le service intentionnel réalise un but et peut offrir une qualité de service (*QoiS-Quality of intentional Service*). Celle-ci peut être une QoiS simple ou une QoiS globale. La QoiS simple concerne le service atomique. Elle correspond à l'ensemble des buts qualité, auxquels le service contribue à satisfaire partiellement. La QoiS globale concerne le service agrégat. Elle correspond à une agrégation de QoiS composantes.

Comme présenté à la Figure 96, la méthodologie de sélection est partagée en trois parties principales : (1) le contexte qualité ; (2) la sélection intentionnelle ; et (3) la configuration intentionnelle.

1. Le contexte qualité est émis par l'agent métier, il correspond à la requête de service. Il correspond au but que le client désire réaliser, ainsi qu'aux différents buts qualité qu'il souhaite satisfaire. Ces buts, l'agent métier peut les trouver à l'aide du référentiel qualité (*cf.* Chapitre 4), lequel référence l'ensemble des buts et métriques connus dans son domaine d'activité. La Section 8.3 présente plus amplement le contexte qualité et sa construction.
2. Le processus de sélection consiste à classer les services intentionnels par rapport au contexte qualité de l'agent métier. Le processus de sélection intervient après la localisation des divers services réalisant le but fonctionnel de l'agent métier client (la localisation de services n'est pas traitée dans cette thèse). Il s'agit alors de sélectionner ces services par rapport au contexte qualité. La Section 8.4 présente ce processus de sélection.

Le processus de configuration est complémentaire au processus de sélection. En effet, si le

résultat de la sélection contient de la variabilité, il est alors possible de configurer le service agrégat. Le processus de configuration permet d'abord de générer les configurations exécutables d'un service agrégat, en considérant les contraintes de dépendance, et ensuite classer ces configurations en fonction du contexte qualité de l'agent métier. La Section 8.5 détaille le processus de configuration.

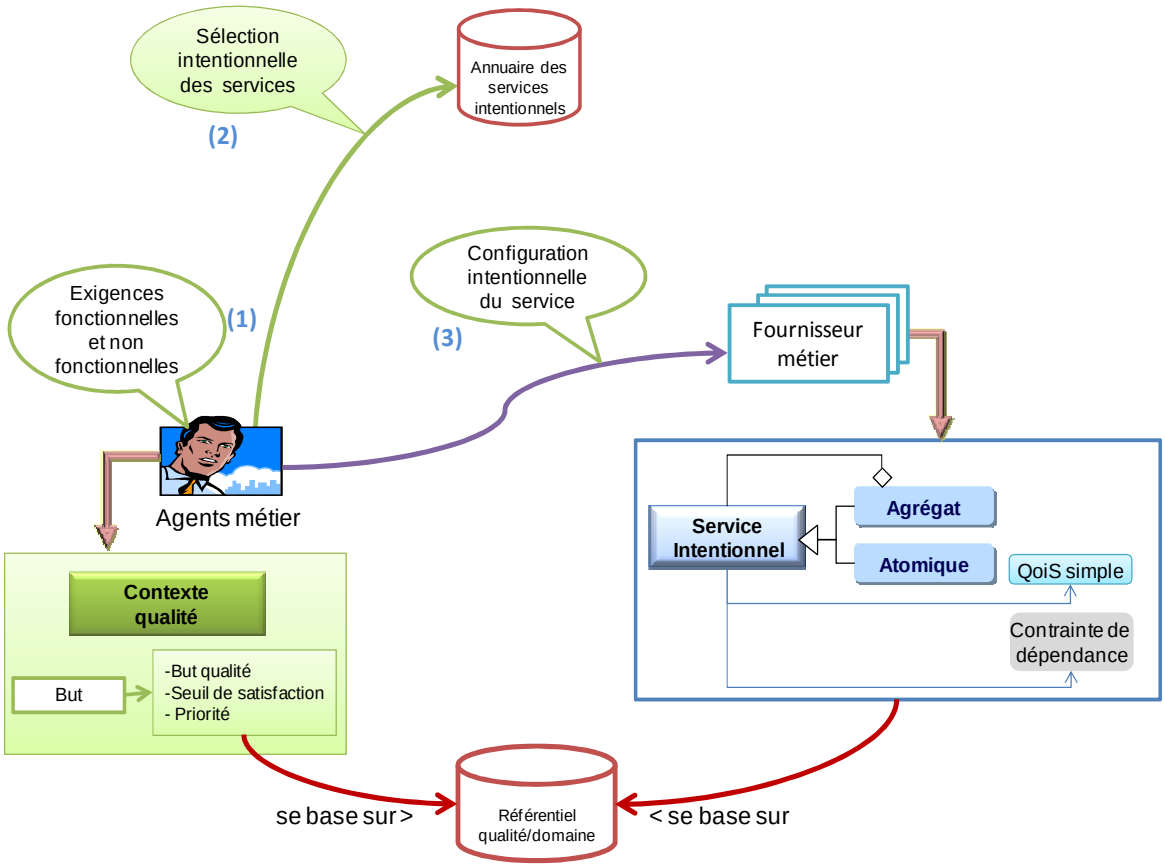

Figure 96. Vue globale de la méthodologie de sélection intentionnelle des services

Les processus de sélection et de configuration se basent sur la méthode multicritères d'aide à la décision TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) (Hwang&Yoon,1981). Cette dernière est utilisée pour classer les services et les configurations les plus proches du contexte qualité de l'agent métier.

8.3 *LE MODELE DU CONTEXTE QUALITE*

Afin de faciliter le processus de sélection nous proposons de modéliser les exigences fonctionnelles et non-fonctionnelles de l'agent métier par un contexte qualité. Ce dernier permet, particulièrement, à l'agent métier de préciser ses préférences en termes de qualité.

Le modèle du contexte qualité est présenté à la Figure 97, en utilisant la notation UML[15]. Trois éléments clés sont présentés dans ce méta-modèle, à savoir le *but*, le *but qualité* et le *contexte qualité*.

En effet, un agent métier, souhaitant réaliser un but particulier (élément *but* de la Figure 97) peut préciser la (ou les) qualité(s) (élément *But qualité* de la Figure 97) qu'ils désirent satisfaire. De plus, l'agent métier peut préciser également le seuil de satisfaction (attribut *Seuil de satisfaction* de la Figure 97) au-delà duquel il accepte toute satisfaction. Il peut aussi définir la priorité qu'il préfère associer à chaque qualité (attribut *Priorité* de la Figure 97).

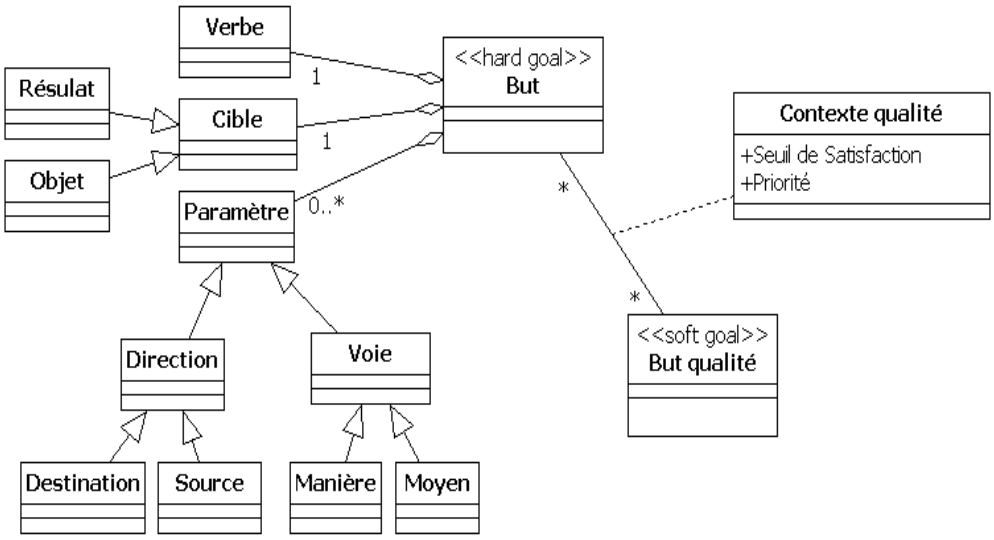

Figure 97. Le modèle du contexte qualité

8.3.1 DEFINITION DES ELEMENTS DU CONTEXTE QUALITE

Nous donnons, dans ce qui suit, une description des différents éléments du modèle du contexte qualité (*cf.* Figure 97), à savoir le but, le but qualité, et le contexte qualité en soi.

8.3.1.1 *LE BUT*

Un but (*cf.* Chapitre 2) exprime une intention, un objectif que l'on souhaite atteindre et que le système doit réaliser. Le but du service intentionnel se base sur l'approche linguistique, initialement développée par Prat (Prat,1997). Dans la perspective de faciliter le processus de localisation des services, nous proposons que le but du contexte qualité se base également sur cette approche. Elle se base sur le fait que la sémantique d'un but est capturée par un verbe et un ou plusieurs paramètres, à savoir la cible, la manière, le moyen, *etc*. A chaque paramètre est associée une fonction sémantique. La même fonction ne peut être associée à deux

[15] Unified Modelling Language: www.uml.org/

paramètres différents (*cf.* Chapitre 4). Par exemple, dans le but $(Payer)_{verbe}$ $(une\ facture)_{objet}(par\ carte\ bancaire)_{moyen}$, le moyen choisi pour réaliser le but *Payer une facture* est la « *carte bancaire* ».

Le but du service intentionnel et le but du contexte qualité sont ainsi formalisés en se basant sur la même approche (*i.e.* par un verbe, une cible et un ensemble de paramètres). Ceci permet de localiser aisément le but exprimé par les agents métier.

8.3.1.2 *LE BUT QUALITE*

Les buts qualité (*cf.* Chapitre 2) du contexte qualité correspondent aux exigences non-fonctionnelles, telles que la sécurité, la performance et la précision, que les agents métier souhaitent satisfaire. Mylopoulos et ses collègues (Mylopoulos et *al.*,1992) ont été les premiers à modéliser les exigences non-fonctionnelles comme des buts qualité à satisfaire partiellement. Le but qualité est défini par une satisfaction subjective, laquelle ne peut être établie de façon précise, vu que sa satisfaction dépend du contexte du client.

Dans le cadre de l'architecture iSOA, nous proposons que l'agent métier se base sur le référentiel qualité (*cf.* Chapitre 5) du domaine qui le concerne pour décider des qualités qu'il souhaite satisfaire. D'une part, ces qualités correspondent aux buts qualité complexes ou simples, lesquels correspondent aux qualités compréhensibles par l'agent métier. D'autre part, ce dernier consulte également le référentiel qualité pour vérifier la sémantique des seuils de satisfaction, afin qu'il établisse ceux qui lui conviennent.

8.3.1.3 *LE CONTEXTE QUALITE*

Le contexte qualité dépend du but et des buts qualité. Il correspond à ce que souhaite l'agent métier. Il permet d'associer des qualités (*i.e.* buts qualité) au but fonctionnel qu'un agent métier souhaite réaliser. Par exemple, un agent métier désire trouver les services lui permettant d'effectuer l'approvisionnement des produits, tout en satisfaisant la Performance et la Fiabilité. Ces derniers ont été sélectionnés par l'agent métier, après consultation du référentiel qualité correspondant au domaine de la gestion des stocks.

Grâce au contexte qualité, l'agent métier peut varier les qualités qu'il souhaite satisfaire, en changeant ses préférences en termes de seuil de satisfaction et de priorité. Ce contexte est utilisé pour sélectionner les services et les configurations qui répondent au mieux aux exigences non-fonctionnelles de l'agent métier.

Le contexte qualité est composé de deux attributs prépondérants, à savoir le seuil de satisfaction et la priorité.

LES SEUILS DE SATISFACTION

Le but qualité se base sur le concept de *satisficing* (*i.e.* accepter la satisfaction) qui consiste à définir un ensemble de seuils de satisfaction et à que l'agent métier accepte tout aboutissement au-delà de ces seuils. Ces derniers correspondent à l'ensemble d'expressions « *Très satisfait* », « *Satisfait* », « *Neutre* », « *Peu satisfait* » et « *Pas du tout satisfait* », notées, respectivement, par les symboles « ++ », « + », « - » et « -- » (*cf.* Chapitre 5). Ainsi, les agents métier peuvent facilement préciser leurs préférences en termes de seuil de satisfaction. En effet, ces seuils ont l'avantage d'être plus facilement compréhensibles aux agents métier que les métriques ou tout autre indicateur quantitatif.

LA PRIORITE

L'agent métier peut souhaiter des buts qualité qui ne peuvent pas être satisfaits simultanément. Par exemple, en règle générale, la Sécurité et la Performance ne peuvent être satisfaites simultanément. En effet, la performance d'un service tend à diminuer lorsque son niveau de sécurité est important. Dans ce cas, l'agent métier doit préciser ses préférences en affectant une priorité à chacun des buts qualité qu'il désire satisfaire. La priorité est le moyen de préciser l'importance d'un but qualité par rapport à d'autres.

Dans notre cas, la priorité est un rang d'importance d'un but qualité, par rapport à l'ensemble de buts qualité que l'agent métier souhaite satisfaire. La priorité est définie par une valeur décimale comprise entre *0.0* et *1.0* de manière à ce que la somme des priorités soit égale à *1.0*. Si une priorité est proche de 0, cela signifie que le but qualité n'est pas prioritaire par rapport aux autres. Si la priorité est proche de 1, cela signifie que le but qualité est extrêmement significatif. Par exemple, si le but qualité Performance a une priorité de *0,7* et le but qualité Sécurité a une priorité de *0,3*, cela signifie alors que la Performance est plus importante que la Sécurité pour le client.

8.3.1.4 *NOTATION DU CONTEXTE QUALITE*

L'agent métier fournit les données relatives à son contexte qualité. Ce dernier correspond au but (noté *B*) que l'agent métier souhaite réaliser, ainsi qu'à l'ensemble des buts qualité (noté *q*) que l'agent métier désire satisfaire. À chacun de ces buts qualité, le seuil de satisfaction désiré (noté *d*) est précisé, ainsi que la priorité (noté *p*), relative à l'importance que donne l'agent métier à ce but qualité *q*. Le contexte qualité est le moyen permettant, à l'agent métier, de varier, pour le même but, les qualités qu'il désire satisfaire, les seuils de satisfaction qu'il souhaite atteindre, ainsi que les priorités qu'il associe à ces qualités.

Soit *n* le nombre de but qualité que l'agent métier désire satisfaire, le contexte qualité est ainsi noté :

$$Cq = B.\{<q_j, dj, pj>\}/\ j=1..n \qquad \textit{notation 1}$$

EXEMPLE

Nous reprenons l'exemple de la gestion de stock. Le client peut fournir le contexte qualité suivant :

Cq = Effectuer l'approvisionnement en produits.{<Performance, +,0,3>,<Fiabilité, +,0,7>}

Ce contexte explicite que le client recherche les services permettant de réaliser le but *Effectuer l'approvisionnement en produits*. Le client désire également satisfaire les qualités de *Performance* et de *Fiabilité* avec des seuils de satisfaction identiques (*i.e.* « + ») et des priorités différentes (*i.e. 0,3* pour la performance et *0,7* pour la fiabilité). Le client privilégie ainsi la fiabilité à la performance.

Tel que défini plus haut (*cf.* Section 8.3.1.2), l'agent métier peut exprimer, dans son contexte qualité, des buts qualité complexes, alors que le fournisseur de service publie une qualité correspondant à des buts qualité simples. Afin de faciliter la sélection et la configuration de services, nous proposons à l'agent métier de consulter le référentiel qualité du domaine (*cf.* Chapitre 5) afin d'effectuer une réécriture du but qualité complexe en un ensemble de buts qualité simples. Comme le montre la Figure 98, le contexte qualité $<q_1,d_1,p_1>$ peut être réécrit en remplaçant le but qualité complexe « q_1 » par tous les buts qualité simples qui le composent, dans le cas de lien d'affinement *Et* entre but qualité complexe et buts qualité simples, à savoir « q_{11} » et « q_{12} ». Dans le cas de lien d'affinement *Ou* entre but qualité complexe et buts qualité simples, l'agent métier peut alors choisir, parmi les buts qualité simples, ceux qu'il souhaite satisfaire (*i.e.* « q_{11} » ou « q_{12} »). Etant donné que l'agent métier souhaite satisfaire « q_1 » avec le seuil de satisfaction « d_1», alors, suivant la valeur de « d_1 » et du lien d'affinement, l'agent peut consulter le Tableau 8 pour décider des seuils de satisfaction avec lesquels il peut satisfaire « q_{11} » et « q_{12} ». Par exemple, si « d_1 = ++ », alors les buts qualité « q_{11} » et « q_{12} » doivent être également satisfaits avec le seuil « ++ » (*cf.* Tableau 8). Il s'agit d'une propagation des seuils de satisfaction vers les buts qualité simple, à partir des buts qualité complexes, en se basant sur ce qui est défini plus haut (*cf.* Tableau 8). Aussi, l'agent métier affecte la priorité « p_1 » pour satisfaire « q_1 ». Il s'agit alors de redistribuer à son souhait p_1 entre q_{11} et q_{12} , de façon à ce que la somme des priorités soit égale à p_1.

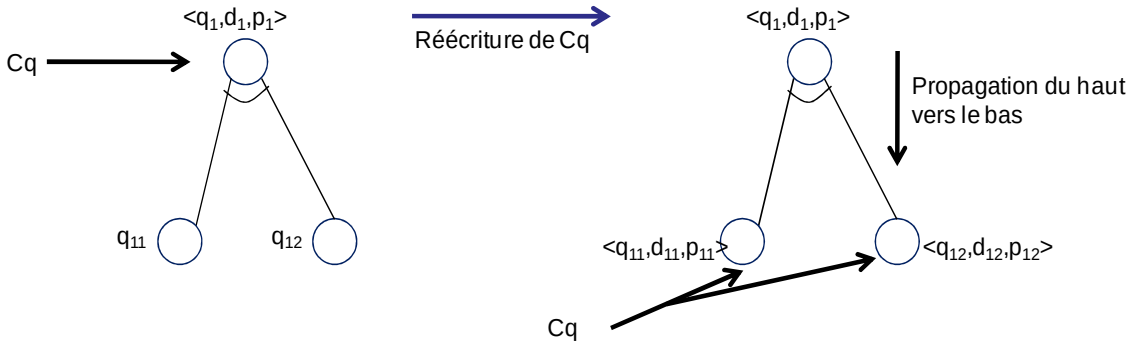

Figure 98. Réécriture du contexte qualité dans le cas de lien d'affinement *Et*

Par exemple, dans le domaine de l'approvisionnement de produits, l'agent métier souhaite satisfaire la *Performance* avec le seuil de satisfaction « + » et la priorité « *0,3* ». En consultant le référentiel du domaine en question, ceci se traduit par satisfaire le but qualité simple *Temps* ou le but qualité simple *Espace*. En consultant le Tableau 8, l'agent peut satisfaire ces buts qualité simples en précisant un seuil « + » à l'un deux et « + », « - » ou « -- » à l'autre. Aussi, l'agent métier peut affecter les priorités « *0,2* » et « *0,1* » aux buts qualité *Temps* et *Espace*.

8.4 *LA SELECTION INTENTIONNELLE DES SERVICES*

Le processus de sélection intentionnelle correspond à un mécanisme d'exécution intentionnelle des services, lequel cherche à guider l'agent métier dans le choix des services répondant le mieux à son contexte qualité (*cf.* Section 8.3.1.3).

Les fournisseurs de services intentionnels publient deux éléments importants pour la sélection intentionnels des services, à savoir : (1) le but fonctionnel que le service réalise ; et (2) la qualité (*QoiS-Quality of intentional Service*) que celui-ci fournit. La QoiS est publiée pour le service atomique et pour le service agrégat. Pour ce dernier, la qualité minimale $QoiS_{min}$ est calculée pour choisir entre les différents services agrégats. Le processus de sélection compare le contexte qualité de l'agent métier aux informations publiées, afin de trouver les services correspondant au mieux aux exigences non-fonctionnelles exprimées par les agents métier. Par ailleurs, le service intentionnel introduit de la variabilité dans la manière d'atteindre le but du service. La variabilité de service s'apparente à une famille de services.

Comme Maximilien et Singh (Maximilien&Singh,2004a), nous faisons la distinction entre la localisation (ou la découverte) et la sélection de service. Ainsi, le processus de sélection intentionnelle se fait en deux étapes : la localisation et la sélection des services intentionnels.

La première étape consiste à localiser ou à découvrir les services qui réalisent le but fonctionnel de l'agent métier (*i.e.* le *B* de *Cq*). Cette localisation consiste à rechercher les

services dont la description (*i.e.* le but du service) possède des similitudes avec *B*. Des travaux concernant la recherche de services Web par les buts existent (Guzélian,2007) (Mirbel&Crescenzo,2009) (Da Silva Santos et *al.*,2009). Dans cette thèse, nous supposons que la localisation des services existe déjà dans la l'architecture iSOA (Aljoumaa et *al.*, 2010) (Aljoumaa et *al.*, 2011).

La deuxième étape consiste à sélectionner parmi les services qui réalisent le but fonctionnel de l'agent métier (*i.e. B* de *Cq*), ceux qui satisfont partiellement les buts qualité de son contexte qualité (*i.e.* q_j de *Cq*). La sélection doit prendre en compte les seuils de satisfaction (*i.e* d_j de *Cq*), ainsi que les priorités (*i.e.* p_j de *Cq*) que l'agent métier précise pour chacune des qualités souhaitées.

La Figure 99 montre un exemple de sélection de services. Dans cet exemple, trois fournisseurs (Fournisseur 1, Fournisseur 2 et Fournisseur 3) publient trois services agrégats ($\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$ à la Figure 99). Ces services appartiennent à la même famille de service : ils réalisent le même but fonctionnel (But à la Figure 99), mais ils offrent des QoiS minimales différentes (notées $QoiS_{min}(\mathbf{S}_1)$, $QoiS_{min}(\mathbf{S}_2)$ et $QoiS_{min}(\mathbf{S}_3)$). Il s'agit alors de sélectionner parmi $\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$ ceux qui correspondent au mieux à un contexte qualité donné. Nous proposons l'usage des méthodes multicritères pour la sélection de ces services en fonction du contexte qualité de l'agent métier. La Section 8.7 détaille la méthode choisie.

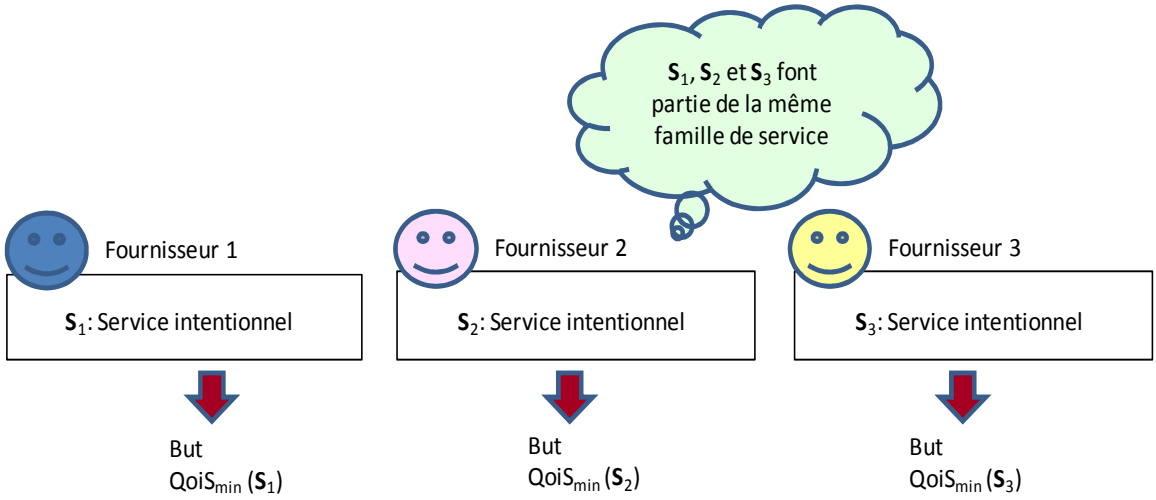

Figure 99. Cas de la sélection des services

8.5 *LA CONFIGURATION INTENTIONNELLE DU SERVICE AGREGAT*

Le service intentionnel agrégat introduit de la variabilité dans la manière d'atteindre le but du service. La variabilité de service s'apparente à une famille de services. Celle-ci permet de mettre en évidence les caractéristiques communes aux services, tout en rendant explicite les variantes. Ces variantes concernent non seulement les différentes manières de réaliser le but

du service, mais aussi de satisfaire partiellement les buts qualité. Ces dernières (*i.e.* les variantes) permettent de répondre aux exigences fonctionnelles et non-fonctionnelles des clients de manière différenciée.

Le processus de configuration intentionnelle correspond à un mécanisme d'exécution intentionnelle du service agrégat, lequel cherche à guider l'agent métier dans le choix successif de variantes à travers les points de variation du service agrégat. La configuration intentionnelle de service doit nécessairement considérer les contraintes de dépendance définies entre les services, afin de garantir l'exécutabilité des configurations du service agrégat (*cf.* Chapitre 6). En effet, le choix d'un ou plusieurs services peut exiger ou exclure le choix d'un autre service, et ainsi entraver l'excécutabilité d'une certaine configuration du service. Donc, il faut considérer les contraintes de dépendance existantes entre les variantes du service agrégat. Les contraintes de dépendance peuvent être de type Exige ou Exclut (voir Chapitre 6) : la contrainte Exige implique qu'un service nécessite la présence d'un autre service dans la configuration de services, tandis que la contrainte Exclut suppose que le choix d'un service écarte la présence d'un autre service dans la dite configuration. La configuration intentionnelle consiste alors à d'abord générer les configurations exécutables du service agrégat, lesquelles satisfont les contraintes de dépendance. Ensuite, ces configurations sont comparées, car elles peuvent potentiellement fournir des qualités différentes.

Dans le cas d'un service agrégat[16], les configurations d'un service peuvent fournir des qualités différentes que celles fournies par le service agrégat lui-même (cette dernière est minimale). Nous proposons de comparer les configurations générées, en calculant leur qualité moyenne $QoiS_{moy}$, telle que définie dans le Chapitre 6. La sélection de services agrégats exploite la qualité minimale publiée par les fournisseurs. Celle-ci correspond à la qualité garantie par ces services. La configuration du service agrégat, quant à elle, calcule la qualité moyenne de chaque configuration exécutable générée. La qualité moyenne permet ainsi d'affiner la comparaison des configurations.

La Figure 100 illustre ce processus de configuration de services. Il s'agit de configurer le service $\mathbf{S}_2$, afin de choisir les configurations de $\mathbf{S}_2$ qui correspondent au mieux à un contexte qualité donné. A partir du service agrégat $\mathbf{S}_2$, plusieurs configurations sont générées : $\mathbf{Conf}_{2.1}$, $\mathbf{Conf}_{2.2}$, …($\mathbf{Conf}_{2.n}$)/ *n* étant le nombre de configurations. Celles-ci réalisent le même but (But à la Figure 100) et sont générées en figeant des choix vis-à-vis des opérateurs d'agrégation, qui représentent les points de variation du service agrégat. Ces configurations doivent

[16] Dans le cas d'un service atomique, il n'existe pas d'autres alternatives que d'accepter la qualité fournie par ce service.

également considérer les contraintes de dépendance, lesquelles permettent de valider l'exécutabilité de ces configurations. Les configurations **Conf**$_i$ offrent des QoiS$_{moy}$ différentes, notées QoiS$_{moy}$ (**Conf**$_{2.1}$), QoiS$_{moy}$ (**Conf**$_{2.2}$), … QoiS$_{moy}$(**Conf**$_n$). Il s'agit alors de choisir parmi **Conf**$_{2.1}$, **Conf**$_{2.2}$, …**Conf**$_n$, celles qui correspondent au mieux à un contexte qualité donné.

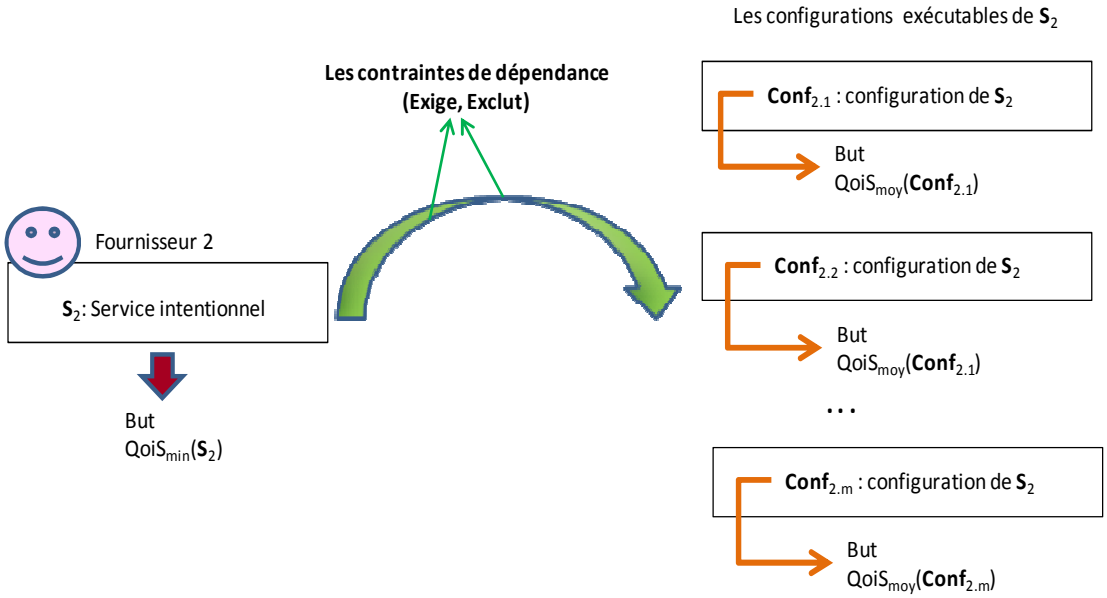

Figure 100. Cas de la configuration des services

La sélection et la configuration intentionnelle doit donc prendre en compte quatre paramètres importants :

1. Choix parmi plusieurs services intentionnels (ou configurations), lesquels réalisent le même but « *B* » ;

2. Différents critères de sélection, lesquels correspondent aux différents buts qualité exprimés dans le contexte qualité de l'agent métier, tels que la sécurité, la précision et la performance ;

3. Des seuils de satisfaction, également exprimés dans le contexte qualité, au-delà desquels l'agent métier est partiellement satisfait ; et

4. Les priorités permettant de préciser les préférences de l'agent métier, concernant des buts qualité contradictoires.

On est alors confronté à un problème de décision pour lequel il existe un ensemble discret d'alternatives représenté par plusieurs services possibles (ou configurations), et plusieurs buts qualité (*i.e.* critères d'évaluation) à prendre en compte pour juger ces services. Généralement, aucun de ces services ne correspond complètement à tous les critères, surtout que ces derniers sont souvent conflictuels. Il s'agit alors de chercher un compromis parmi les services (ou

configurations) existants, en combinant les différents critères, afin de pourvoir classer les services les plus proches du contexte qualité de l'agent métier.

Nous pensons comme (Zeng et *al.*,2003) (Wang et *al.*,2006) (Herssens et *al.*,2008) (Ma et a*l.*,2009) que les méthodes d'aide à la décision multicritères sont un bon moyen de résoudre ce type de problème. Particulièrement, à l'instar de (Yamamoto&Saeki,2008), nous proposons d'utiliser la méthode TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) (Hwang&Yoon,1981) (Zavadskas,1986), afin de guider l'agent métier dans son choix. Nous optons pour la méthode TOPSIS pour sa simplicité et pour la méthode de calcul qu'elle adopte (*i.e.* la distance euclidienne de la meilleure et de la pire solution). Notamment, nous avons opté pour TOPSIS, car il est possible de la personnaliser. En effet, au lieu de sélectionner les alternatives ayant la plus courte distance de la meilleure solution, nous proposons d'adapter TOPSIS, afin de sélectionner les services (ou les configurations) ayant la plus courte distance du contexte qualité de l'agent métier. Le contexte qualité correspond à ce que souhaite l'agent métier en termes de but qualité, de seuil de satisfaction et de priorité.

8.6 *L'AIDE A LA DECISION MULTICRITERES*

Zeleny (Zeleny,1982) commence son livre "*Multiple Criteria Decision Making*" par la déclaration suivante :

"*It has become more and more difficult to see the world around us in a unidimensional way and to use only a single criterion when judging what we see*"

En effet, l'analyse monocritère repose sur un seul critère, par exemple la minimisation des coûts, la maximisation des profits ou la minimisation d'une distance parcourue. Cependant, elle n'est souvent pas le reflet de la réalité, car il devient de plus en plus difficile de voire le monde qui nous entoure de façon unidimensionnelle et d'utiliser seulement un critère pour juger et décider de ce que nous voyons. Par exemple, un chef d'entreprise peut effectivement rechercher la maximisation des profits, mais il doit également prendre en compte la situation de l'emploi et respecter l'environnement.

Par conséquent, l'analyse monocritère devient multicritère lorsque le problème comporte plusieurs critères, souvent contradictoires. L'analyse multicritère est un outil permettant d'aider les décideurs à prendre une décision ou à évaluer plusieurs options ou solutions, dans des situations où aucune possibilité n'est parfaite. Elle est adoptée dans plusieurs champs d'application pour résoudre différents types de problèmes, tels que les problèmes de gestion financière (le choix des projets d'investissement, l'octroi de crédits bancaire…), les problèmes de gestion de personnel (le recrutement de candidats, la mise en place d'un

système d'évaluation…) et d'autres problèmes aussi complexes (le dépouillement d'un appel d'offre, la localisation d'une nouvelle installation…) (Caillet,2003).

Les méthodes multicritères considèrent alors un ou plusieurs décideurs qui sont face à un problème et qui disposent pour cela de plusieurs solutions possibles (appelées aussi options ou actions). Le décideur doit prendre en compte plusieurs critères (ou des points de vue) pour juger ces solutions, mais aucune solution ne se dégage du lot (*i.e.* aucune solution n'est la plus performante sur tous les critères). De surcroît, les critères peuvent être conflictuels.

Prenons un exemple anodin (Caillet,2003) : un consommateur veut acheter une nouvelle voiture. Les critères dont il prend compte pour faire son choix sont le prix, la puissance et la beauté de la voiture. Il a le choix entre plusieurs modèles de toutes marques. A priori, les critères puissance et prix sont conflictuels, car on paie plus pour avoir plus de puissance. De plus, entre la Fiat et la Ferrari, de nombreuses autres voitures (solutions possibles) sont envisageables pour le consommateur. Parmi toutes ces possibilités, le choix est difficile et l'analyse multicritère peut aider le consommateur à effectuer son choix.

Ainsi, les méthodes multicritère d'aide à la décision permettent de définir des modèles permettant d'améliorer le processus décisionnel dans les situations de choix où aucune solution n'est parfaite et où différents critères entrent en conflit. L'idée de base est de combiner différents critères, autant quantitatifs que qualitatifs, entrant en compte ; leur attribuer un poids lié à leur importance relative ; évaluer chaque solution par rapport à tous les critères ; agréger ces résultats ; et finalement classer les solutions envisageables. L'analyse multicritère est, en quelque sorte, une manière de chercher un compromis parmi plusieurs solutions.

Nous pensons que l'aide à la décision multicritères est le moyen permettant de sélectionner, parmi les services (ou les variantes) réalisant le but de l'agent métier, ceux dont les qualités correspondent aux buts qualité que l'agent métier souhaite satisfaire. Ces buts qualité, étant multiples, représentent des critères de choix souvent conflictuels. L'analyse multicritères permet alors de trouver un compromis satisfaisant entre l'offre de services et les préférences de l'agent métier.

8.6.1 LES METHODES MULTICRITERES D'AIDE A LA DECISION

L'aide à la décision multicritère (*MCDM-Multi-Criteria Decision Making*) est une discipline de la recherche opérationnelle, laquelle traite des problèmes de décision en présence d'un certain nombre de critères. Généralement, MCDM est classée (Zimmermann,1991) en « aide à la décision multi-objectif (*MODM-Multi-Objective Decision*

Making) » et « aide à la décision multi-attribut (*MADM- Multi-Attribute Decision Making*) ». MODM étudie les problèmes dans lesquels l'espace de décision est continu, tandis que MADM se concentre sur des problèmes dans lesquels l'espace de décision est discret. Dans ce cas, l'ensemble des alternatives de décision sont prédéterminées. Dans la littérature (Triantaphyllou et *al.*,1998), les termes de MADM et MCDM sont souvent confondus, ils sont utilisés pour désigner la même classe de modèles.

Dans ce rapport de thèse, nous sommes confrontés à un ensemble connu de services (ou de variantes), c'est-à-dire ceux qui réalisent le but de l'agent métier. Nous utilisons donc une méthode MADM pour résoudre notre problème. Nous utilisons dans la suite de ce rapport le terme de MCDM pour désigner MADM.

8.6.2 LES CONCEPTS DES METHODES MCDM

Les méthodes MCDM se partagent un ensemble de concepts, à savoir (Triantaphyllou et *al.*,1998) (Triantaphylou,2000) les alternatives, les attributs multiples, ou encore la matrice de décision.

ALTERNATIVES

Les alternatives représentent les différents choix d'action disponibles au décideur. Généralement, l'ensemble des solutions est supposé fini. Elles doivent être présélectionnées, priorisées et finalement classées.

ATTRIBUTS MULTIPLES

Chaque problème MCDM est associé à de multiples attributs, appelés également dans la littérature "buts" ou "critères de décision". Les attributs représentent les différentes dimensions à partir desquelles les alternatives peuvent être vues. Dans le cas d'un nombre important d'attributs (*i.e.* plus qu'une douzaine), les attributs peuvent être arrangés d'une façon hiérarchique. Ainsi, quelques attributs peuvent être des attributs majeurs. Chaque attribut majeur peut être associé à plusieurs sous-attributs. De façon similaire, chaque sous-attribut peut être associé à des sous-sous-attributs différents, et ainsi de suite. Malgré le fait que quelques méthodes MCDM considèrent explicitement une structure hiérarchique, la majorité d'entre elles considère un seul niveau d'attributs (Triantaphylou,2000).

CONFLIT ENTRE ATTRIBUTS

Étant donné que les différents attributs représentent les différentes dimensions de l'alternative, les uns peuvent entrer en conflit avec les autres. Par exemple, le coût peut entrer en conflit avec les bénéfices, *etc*.

UNITES INCOMMENSURABLE

Différents attributs peuvent être associés à différentes unités de mesure. Par exemple, dans le cas de l'achat d'une voiture d'occasion, les attributs « coût » et « kilométrage » peuvent être mesurés, respectivement, en termes d'euros et de milliers de kilomètres.

POIDS DE DECISION

La majorité des méthodes MCDM propose d'attribuer aux attributs des poids désignant leurs importances. Généralement, ces poids sont normalisés de telle sorte que leur somme doit être égale à *1*.

MATRICE DE DECISION

Un problème de MCDM peut être facilement exprimé sous forme d'une matrice. Une matrice de décision **A** est une matrice ($m \times n$) dans laquelle l'élément a_{ij} indique la performance de l'alternative A_i lorsque celle-ci est évaluée selon les critères de décision C_j, (pour $i = 1, 2,3,..., m$, et $j = 1, 2,3,..., n$). Il est également supposé que le décideur ait déterminé les poids de la performance relative des critères de décision (noté W_j, pour $j = 1, 2,3,..., n$). Ces informations sont illustrées à la Figure 101.

	Critères				
Alternatives	C_1	C_2	…	C_j …	C_n
	W_1	W_2	…	W_j …	W_n
A_1	a_{11}	a_{12}		a_{1j}	a_{1m}
A_2	a_{21}	a_{22}		a_{2j}	a_{2m}
…	…				
A_i	a_{i1}	a_{i2}		a_{ij}	a_{im}
…	…				
A_m	a_{m1}	a_{m2}		a_{mj}	a_{mn}

Figure 101. Représentation d'une matrice de décision

Un problème MCDM peut être donc défini comme suit : soit $A = \{Ai, pour\ i = 1, 2,3,..., m\}$ l'ensemble fini des alternatives de décision et $G = \{g_i, pour\ j = 1, 2,3,..., n\}$ l'ensemble fini des attributs selon lesquels la pertinence d'une action est jugée. Il s'agit de déterminer

l'alternative optimale *A** ayant la plus grande pertinence par rapport à tous les attributs g_i (Zimmermann,1991). Généralement, les attributs g_i sont appelés critères de décision, ou juste critères (les alternatives doivent être jugées ou évaluées en fonction de ces attributs).

8.6.3 LES ETAPES D'UNE MCDM

Dans une méthode MCDM, la recherche de la solution la plus adéquate possible peut se faire en suivant une démarche composée de 5 étapes principales, à savoir :

1. Dresser la liste des solutions (ou alternatives) possibles ou envisageables. Cette liste n'est pas exhaustive et définitive (la suppression et l'ajout de solutions restent possibles) ;
2. Dresser la liste des critères à prendre en considération lors de l'évaluation ;
3. Pondérer les critères. Un critère peut être plus important qu'un autre. Cette importance relative est exprimée par un nombre appelé poids.
4. Evaluer (ou juger) chacune des solutions en fonction de chacun des critères et de leur poids respectifs. L'ensemble des évaluations est présenté dans un tableau à double entrée, la matrice de décision, dans laquelle chaque ligne représente une solution et chaque colonne un critère ; et
5. Agréger (ou confronter) les résultats des évaluations, afin de désigner la solution qui obtient les meilleures évaluations (ou le meilleur classement).

8.6.4 LES DOMAINES D'APPLICATION DES METHODES MCDM

L'aide à la décision multicritères joue un rôle crucial dans la vie courante. Ainsi, les méthodes MCDM sont appliquées dans plusieurs domaines.

Tille (Tille,2000) recommande l'utilisation de la méthode EleCTRe dans le cadre de projets routiers. Thomas et ses collègues (Saaty et *al.*,2007) proposent d'utiliser AHP (*Analytic Hierarchy Process*) pour résoudre le problème d'allocation de ressources humaines. Dans le domaine du bâtiment, plusieurs études (Fang et *al.*,2004) (Kartam et *al.*,2004) (Merrick&Harrald,2007) (Liaudanskiene et *al.*,2009) ont été menées pour évaluer les solutions de sureté dans le processus de construction.

Plusieurs travaux (Wang et *al.*,2006) (Xiong&Fan,2007) (Ma et *al.*,2009) ont utilisé les méthodes d'aide à la décision multicritères dans la sélection des services Web. Yamamoto et Saeki (Yamamoto&Saeki,2007) (Yamamoto&Saeki,2008) ont eux aussi adopté la méthode TOPSIS pour évaluer les diverses alternatives d'un graphe de buts afin de sélectionner celle qui est la plus proche de l'alternative idéale.

8.6.5 LES TYPOLOGIES DES METHODES MCDM

Les spécialistes de l'aide à la décision multicritère s'accordent sur trois grandes familles de méthodes MCDM (Schärlig,1985) (Vincke,1989)(Roy,1992) (Roy&Bouyssou,1993), à savoir les méthodes basées sur la théorie de l'utilité multi-attributs, les méthodes de sur-classement et les méthodes interactives.

Les méthodes basées sur la théorie de l'utilité multi-attributs : approche du critère unique de synthèse évacuant toute incomparabilité. Ces méthodes sont développées par les tenants de l'« *école nord-américaine* ». Elles consistent à agréger les différents points de vue en une fonction unique qu'il s'agit ensuite d'optimiser. Les travaux relatifs à cette famille étudient les conditions mathématiques d'agrégation, les formes particulières de la fonction agrégeante et les méthodes de construction de ces fonctions. Les principales méthodes appartenant à cette famille sont : WSM (*Weight Sum Method*) (Fishburn,1967), WSP (*Weight Product Method*) (Miller&Starr,1969), AHP (*Analytic Hierarchy Process*) (Saaty,1977) (Saaty,1990) (Saaty&Vargas,2001) et TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) (Hwang&Yoon,1981).

Bien que rigoureuse, ces méthodes peuvent écarter parfois quelques solutions, pourtant optimales, sur la base de compensation possible entre critères, la moyenne pondérée de notes est l'exemple le plus connu de ces techniques (Ben Mena,2000). Toutefois, ces méthodes fournissent des résultats clairs et elles s'avèrent selon (Schärlig, 1985) intéressantes ou tout simplement les seules utilisables.

Les méthodes de sur-classement : approche de sur-classement de synthèse acceptant l'incomparabilité. Ces méthodes sont développées par les tenants de l'« école européenne». Elles visent en première étape à construire une relation appelée relation de sur-classement, qui représente les préférences établies du décideur, compte tenue de l'information dont il dispose. Les méthodes de cette famille introduisent des seuils d'indifférence et de préférence au niveau de chacun des critères avant de construire la relation de sur-classement. Cette relation n'est donc en général ni complète, ni transitive. La seconde étape consiste à exploiter la relation de sur-classement en vue d'aider le décideur à résoudre son problème.

Ces méthodes sont caractérisées par un bon degré de pragmatisme. En comparaison des méthodes précédentes (*i.e.* méthodes basées sur la théorie de l'utilité multi-attributs**)**, les résultats obtenus sont parfois peu clairs, car ils sont basés sur une analyse du graphe des relations qui est difficile et complexe. De plus, le nombre d'opérations de comparaisons à réaliser sur chaque paire de variantes peut se révéler considérable en présence de nombreuses

variantes (pour n variantes, on a n (n-1) comparaisons à réaliser). Les principales méthodes de cette famille sont : Electre (*Elimination and Choice Translating Reality*) (Benayoun, et *al.*,1966) (Roy B., 1968), Prométhée (*Preference ranking organisation methode for enrichment evaluations*) (Brans et *al.*,1984) et Naiade (Munda,1995).

Les méthodes interactives : approche de jugement local interactif avec itération essai-erreur. Les deux méthodes précédentes peuvent se révéler lourdes à utiliser en présence d'un ensemble *V* continu. Les méthodes interactives proposent alors de procéder à une exploration locale en fixant tout d'abord une solution de départ correspondant à une variante initiale qui est aussi bonne que possible. Ensuite, on regarde dans l'ensemble des variantes proches de la variante initiale s'il n'existe pas une variante qui soit meilleure. Si c'est le cas, cette variante devient la variante initiale d'un nouveau processus de recherche. On procède ainsi par itérations.

Ces jugements locaux mettent en jeu un petit nombre de variantes en renonçant à une vision globale du problème posé. Il est ainsi tentant de vouloir augmenter le nombre d'itérations, de manière à limiter le risque d'« oublier » une variante qui pourrait s'avérer intéressante. Ces méthodes sont aussi d'un contenu théorique ardu, ce qui fait que le décideur doit avoir une totale confiance envers l'analyste. Les principales méthodes de cette famille sont STEM (Benayoun et *al.*,1971) et les variantes interactives d'UTA ou du *goal-programming* (Masud&Hwang,1981).

Le Tableau 23 résume les avantages et les inconvénients des trois familles de méthodes MCDM, à savoir les méthodes basées sur la théorie de l'utilité multi-attributs, les méthodes de sur-classement et les méthodes interactives.

Tableau 23. Résumé des méthodes MCDM

Type de méthodes	Avantage	Inconvénient
Les méthodes basées sur la théorie de l'utilité multi-attributs	- rigoureuse - résultats clairs	- solutions écartées
Les méthodes de sur-classement	- bon degré de pragmatisme	- résultats parfois peu clairs - nombre de comparaisons considérable
Les méthodes interactives	- ensemble continu de variantes	- pas de vision globale - contenu théorique ardu

8.6.6 LE CHOIX D'UNE METHODE MULTICRITERES D'AIDE A LA DECISION

Il n'existe pas de méthodes d'aide multicritère à la décision qui soit parfaite et idéale pour chaque cas donné. Le choix de la méthode à utiliser est en soi une problématique qui dépend du décideur, des solutions considérées, du résultat attendu, des objectifs fixés à l'aide à la décision, *etc*.

Cette thèse de doctorat s'intéresse à déterminer la méthode MCDM la mieux adaptée à la sélection et à la configuration intentionnelle des services, en fonction du contexte qualité de l'agent métier. Il s'agit pour cela de proposer une méthode multicritères d'aide à la décision qui :

- considère un ensemble fini de services intentionnels (ou de configurations exécutables), ceux réalisant le but « *B* » ; et
- prend en compte un contexte qualité de l'agent métier, correspondant aux différents critères de sélection, aux seuils de satisfaction au-delà desquels l'agent métier considère qu'il est partiellement satisfait, et les priorités permettant de préciser les préférences des agents métier pour chaque critère.

De plus, cette méthode doit être rigoureuse, facile à utiliser et à réaliser, et surtout elle doit permettre de fournir des résultats compréhensibles aux acteurs non techniques. Elle doit être facile à appliquer au cas de la sélection intentionnelle des services et possèdera un retour d'expérience positif, comparée aux autres méthodes multicritère, utilisées actuellement dans la sélection des services

À partir de la synthèse des caractéristiques des différentes familles de méthodes MCDM, présentée au Tableau 23, notre choix se porte sur la méthode TOPSIS. Elle a été développée par Hwang et Yoon (Hwang&Yoon,1981) et a fait l'objet de nombreuses applications (Schinas,2007) (Yamamoto&Saeki,2008).

Selon Shinas (Schinas,2007), la méthode TOPSIS offre plusieurs avantages aux décideurs, particulièrement au non académiciens, lesquels ne considèrent pas les problèmes théoriques : la méthode TOPSIS fournit une solution euclidienne, c'est-à-dire facile à concevoir et tous les calculs se font dans un temps polynomial. Comme Electre, TOPSIS permet de juger chaque solution par rapport à la meilleure et à la pire solution (les variantes de références dans Electre), mais TOSPSIS est plus simple à mettre en œuvre. Enfin, les décideurs peuvent comprendre plus facilement la déviation d'une solution par rapport à la meilleure et à la pire solution.

La méthode TOPSIS permet de juger chaque solution vis-à-vis de la meilleure et de la pire solution et ceci indépendamment des autres solutions. On peut ainsi rajouter une solution dans l'analyse et rapidement l'étudier sans modifier les résultats déjà obtenus. Elle est facile d'adaptation, car on peut remplacer la meilleure et la pire solution par le contexte qualité et la négation du contexte qualité de l'agent métier, sans que cela puisse affecter sensiblement la méthode.

Néanmoins, ce choix n'est pas le seul applicable bien entendu et il peut être discuté sans que cela remette en cause l'intérêt de la méthode TOPSIS.

8.7 *LA METHODE TOPSIS*

La méthode multicritères d'aide à la décision TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) (Hwang&Yoon,1981) (Zavadskas,1986) est une alternative de la méthode EleCTRe. Le concept de base de cette méthode est que les alternatives sélectionnées doivent avoir, dans un espace géométrique, la plus courte distance euclidienne de la solution idéale et la plus grande distance de la pire solution (*i.e.* négation de la solution idéale). TOPSIS suppose que chaque attribut a une tendance à accroitre ou à décroitre. Ainsi, il est facile de définir les solutions artificielles, dites la solution idéale (qui contient les meilleures valeurs de chaque attribut) et la pire solution (qui contient les pires valeurs de chaque attribut). La distance Euclidienne est utilisée pour calculer la proximité (ou la similitude) relative à la solution idéale. Ainsi, l'ordre de préférence des solutions est donné en les comparant à ces distances relatives.

La méthode TOPSIS suppose donc deux alternatives artificielles, à savoir :

- l'alternative idéale (« *ideal alternative* ») : c'est l'alternative qui possède la meilleure évaluation (ou valeur) par rapport à tous les critères considérés ; et
- la pire alternative (« *negative ideal alternative* ») : c'est l'alternative qui possède la pire évaluation (ou valeur) par rapport à tous les critères considérés.

Les alternatives artificielles sont construites à partir des valeurs des attributs des alternatives existantes, en prenant en compte, respectivement, les meilleures et les pires valeurs de chacun des attributs. La méthode TOPSIS consiste donc à sélectionner l'alternative la plus proche de l'alternative idéale et la plus loin de la pire alternative.

La méthode TOPSIS, tout comme les autres méthodes MCDM, se base sur la matrice de décision (ou d'évaluation) pour effectuer l'agrégation des résultats des évaluations. La représentation d'une matrice de décision (ou matrice d'évaluation) est présentée à la Figure 101.

La matrice de décision, telle que définie à la Figure 101, est composée des éléments suivants :

- Ensemble des alternatives potentielles : $A= \{a_1, a_2, \ldots, a_i, \ldots, a_m\}$, tel que m est le nombre d'alternatives potentielles (a_i où $i=1,2,\ldots,m$) ;
- Différents critères d'évaluation : $C= \{c_1, c_2, \ldots, c_j, \ldots, c_n\}$, tel que n est le nombre des critères d'évaluation (c_j où $j=1,2,\ldots,n$) ;
- Poids des critères $= \{w_1, w_2, \ldots, w_j, \ldots, w_n\}$, tel que n est le nombre de critères (w_j où $j=1,2,\ldots, n$) ;
- Évaluations (ou jugements) : $E_{ij} = \{e_{11}, e_{12}, \ldots, e_{1j}, e_{21}, e_{22}, \ldots, e_{2j}, e_{i1}, e_{i2}, \ldots, e_{ij} \ldots, e_{mn}\}$, tel que e_{ij} représente l'évaluation de d'alternative a_i selon le critère c_j (e_{ij} où $i=1,2,\ldots,m, j=1,2,\ldots,n$).

La méthode TOPSIS suppose donc qu'il existe m alternatives ($a_{i=1\ldots m}$) et n critères d'évaluation ($c_{j=1\ldots n}$). Soit e_{ij} l'évaluation (*i.e.* la valeur) de chaque alternative par rapport à chaque critère, et soient $M = (e_{ij})$ la matrice d'évaluation de ($m \times n$) éléments, J l'ensemble des critères avantageux (*i.e.* meilleur quand la valeur est grande, tel que la rentabilité) et J' l'ensemble des critères désavantageux (*i.e.* meilleur quand la valeur est petite, tel que le temps de réponse), la méthode TOPSIS suit une démarche composée de plusieurs étapes principales. Ci-dessous la description des différentes étapes.

Étape 1 : construire la matrice normalisée

Les valeurs des différents critères sont souvent exprimées dans des unités différentes (euro, kilomètre, kilogramme, *etc.*). La normalisation permet d'uniformiser ces valeurs, afin de pouvoir les comparer. La normalisation des valeurs produit la matrice R et elle se fait comme suit :

$$r_{ij} = e_{ij} / \sum e^2_{ij}$$
$$i=1 \ldots m; j=1 \ldots n$$

Étape 2 : construire la matrice normalisée pondérée

Soit w_j où $j=1,2,\ldots, n$ le poids associé à chacun des critères. Il s'agit de multiplier chaque colonne de la matrice normalisée par le poids correspondant. La pondération des valeurs normalisées produit la matrice V. Celle-ci se calcule comme suit :

$$v_{ij} = r_{ij} \times w_j$$
$$i=1 \ldots m; j=1 \ldots n$$

Étape 3 : Définir les solutions artificielles

L'alternative idéale est généralement une alternative fictive. Elle est définie comme suit :

$$A^* = \{v_1^*, \ldots, v_n^*\} \qquad v^* = \{max\ (v_{ij})\ si\ \ j \in J\ ;\ \ min\ (v_{ij})\ si\ \ j \in J'\}$$

La pire alternative est généralement, aussi, une alternative fictive. Elle est déterminée comme suit :

$$A' = \{v_1', \ldots, v_n'\} \qquad v' = \{\ min\ (v_{ij})\ si\ \ j \in J\ ;\ \ max\ (v_{ij})\ si\ \ j \in J'\ \}$$

Sachant que *J* est l'ensemble des critères avantageux (*i.e.* meilleur quand la valeur est grande) et *J'* est l'ensemble des critères désavantageux (*i.e.* meilleur quand la valeur est petite).

Étape 4 : calculer les distances Euclidiennes.

La méthode TOPSIS propose de calculer la distance Euclidienne de chaque alternative a_i (où $i=1,2,\ldots, m$) à l'alternative idéale, comme suit :

$$S_i^* = [\ \Sigma (v_j^* - v_{ij})^2\ ]^{½} \qquad i = 1, \ldots, m$$

La distance euclidienne de a_i (où $i=1,2,\ldots, m$) à la pire alternative est calculée comme suit :

$$S'_i = [\ \Sigma (v_j' - v_{ij})^2\ ]^{½} \qquad i = 1, \ldots, m$$

Étape 5 : Calculer la proximité (ou la similarité) relative.

La proximité relative $C^*_{i=1..m}$ permet de comparer les différentes alternatives par rapport aux alternatives artificielles *A** et *A'*. Il est considéré comme étant le degré de préférence des *m* alternatives. Le calcul de $C^*_{i=1..m}$ se fait comme suit :

$$C_i^* = S'_i / (S_i^* + S'_i) \qquad i = 1, \ldots, m \qquad 0 < C_i^* < 1$$

Il s'agit de maximiser $C^*_{i=1..m}$: les alternatives dont les $C^*_{i=1..m}$ sont proches de *1* sont les plus proches de l'alternative idéale et les plus éloignées de la pire alternative. Par conséquent, lorsque $C^*_{i=1..m}= 1$ alors $a_i = A^*$, c'est-à-dire qu'il s'agit de la meilleure alternative. Si $C^*_{i=1..m} = 0$ alors $a_i = A'$, c'est-à-dire qu'il s'agit de la pire alternative.

Étape 6 : classer les alternatives selon le degré de préférence.

La meilleure alternative peut désormais être décidé en fonction du degré de préférence $C^*_{i=1..m}$: les meilleures alternatives seront celles qui ont, d'une part, la plus courte distance de l'alternative idéale (*i.e.* $S^*_{i=1..m}$), et d'autre part, la plus longue distance de la pire alternative (*i.e.* $S'_{i=1..m}$).

La Figure 102 présente un exemple qui montre le positionnement de la meilleure solution (celle qui maximise C^*_i) selon les critères de Coût et d'Efficacité. L'alternative a_1 possède le

pire coût, alors que l'alternative a_2 présente le meilleur coût, car plus le coût est important moins l'alternative est intéressante, contrairement à l'efficacité. Ainsi, l'alternative idéale est celle qui présente la plus grande valeur d'efficacité et la plus petite valeur de coût, alors que la pire alternative est celle qui possède la plus petite valeur d'efficacité et la plus grande valeur de coût. La meilleure solution est celle qui possède la plus courte distance de l'alternative idéale et la plus longue distance de la pire alternative, *i.e.* la solution a_i à la Figure 102. La Figure 103 résume l'algorithme de la méthode TOPSIS.

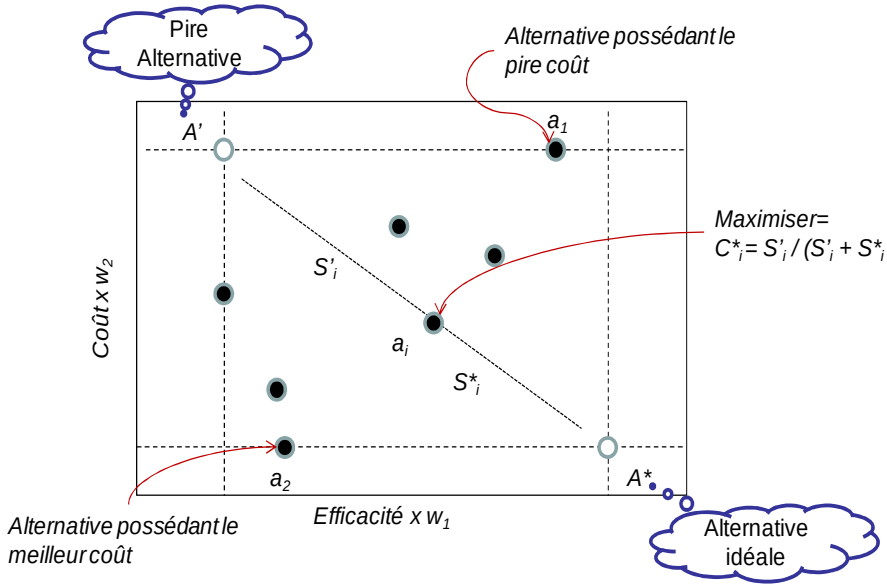

Figure 102. La méthode TOPSIS (Yamamoto&Saeki,2008)

TOPSIS()

Entrée
m : nombre d'alternatives
n : nombre de critères
a[i] : les alternatives possibles
c[j] : les critères d'évaluation
w[j] : les poids pour chaque critère d'évaluation $c_{j/j=1\ldots n}$
e[i,j] : évaluation de chaque alternative a[i] suivant les critères c[j]
M : matrice de décision de ($m \times n$) éléments
R : matrice de décision normalisée
V : matrice de décision normalisée pondérée

Sortie
ListeAlternative[i] : classement descendant des alternatives dont C^* est proche de 1

DEBUT
J= {$c_{j/j=1\ldots s}$} : s est le nombre des meilleurs $c_{j/j=1\ldots n}$ quand la valeur est grande
J'= {$c_{j/j=1\ldots t}$} : t est le nombre des meilleurs $c_{j/j=1\ldots n}$ quand la valeur est petite
M[i,j]=ConstruireM (a[i],**m**,c[j],n*)*
R[i,j]=ConstruireR (*M*)
V[i,j]=ConstruireV (*R*)
A*[j]=CalculerA*(V[i,j],J,J')

```
   A'[j]=CalculerA'(V[i,j],J,J')
   S*[i]=CalculerS* (V[i,j],A*[j])
   S'[i]=Calculer (V[i,j],A'[j])
   C*[i]=Calculer (S*[i],S'[i])
   ListeAlternative[i]=Classer(C*[i])

   ConstruireM(a[i],m,c[j],n)
   { pour i=1,m
         pour j=1,n
                  e[i,j]= evaluation(a[i],c[j])
   }
   ConstruireR(M)
   { pour i=1,m
            pour j=1,n
                  r[i,j]= e[i,j]/(Σ(e[i,j])²)
   }
   ConstruireV(R)
   { pour i=1,m
            pour j=1,n
                  v[i,j] = w[j] × r[i,j]
   }
   CalculerA* (v[i,j],J,J')  // alternative idéale
   { pour i=1,m
            pour j=1,n
            v*[j] ={max (v[i,j]) si j ∈ J; min(v[i,j]) si j∈ J'}
     A*[j]= v*[j]
   }
   CalculerA' (v[i,j],J,J')  // pire alternative
   { pour i=1,m
            pour j=1,n
            v'[j]={min (v[i,j]) si j∈J; max (v[i,j]) si j∈J'}
     A'[j]= v'[j]
   }
   CalculerS* (v[i,j],A* )  // distance de l'alternative idéale
   { pour i=1,m
            pour j=1,n
               S*[i] =[ Σ (v*[j] – v[i,j])² ]½
   }
   CalculerS' (v[i,j],A')  // distance de pire alternative
   { pour i=1,m
            pour j=1,n
                S'[i] =[ Σ (v'[j] – v[i,j])² ]½
   }
   CalculerC* (S'[i],S*[i]) // proximité relative ou degré de préférence
   { pour i=1,m
         C*[i] = S'[i]/(S*[i] + S'[i])   // 0 <  C*[i] < 1
   }
   Classer (C*[i])
   { ListeAlternative[i] = les alternatives classées dans un ordre
     descendant par rapport à C*[i]
   }

FIN
```

Figure 103. Algorithme de la méthode TOPSIS

Nous présentons, dans ce qui suit, l'adaptation de la méthode TOPSIS à la sélection intentionnelle des services, suivant le contexte qualité du client.

8.7.1 ADAPTATION DE TOPSIS A LA SELECTION INTENTIONNELLE DES SERVICES

Nous adaptons la méthode TOPSIS pour sélectionner les services intentionnels les plus proches du contexte qualité de l'agent métier. Comme le montre la Figure 104, nous adaptons TOPSIS pour sélectionner parmi les services intentionnels $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{S}_3$…$\mathbf{S}_{i/i=1..m}$ (*m* est le nombre de services) ceux qui se rapprochent le plus du contexte qualité de l'agent métier, à savoir *B.* $\{< q_j, d_j, p_j >\}$.

Nous définissons dans ce qui suit l'adaptation de chaque étape de la méthode TOPSIS (*cf.* Section 8.7) à la sélection intentionnelle des services, selon le contexte qualité de l'agent métier. Avant d'adapter les différentes étapes, nous proposons d'abord de construire la matrice de décision correspondant à la sélection des services de la Figure 104. Les alternatives, les critères d'évaluation et l'évaluation des alternatives de TOPSIS sont remplacés, respectivement, par les services intentionnels, les buts qualité que l'agent métier souhaite satisfaire et la qualité fournie par ces services, concernant ces buts qualité.

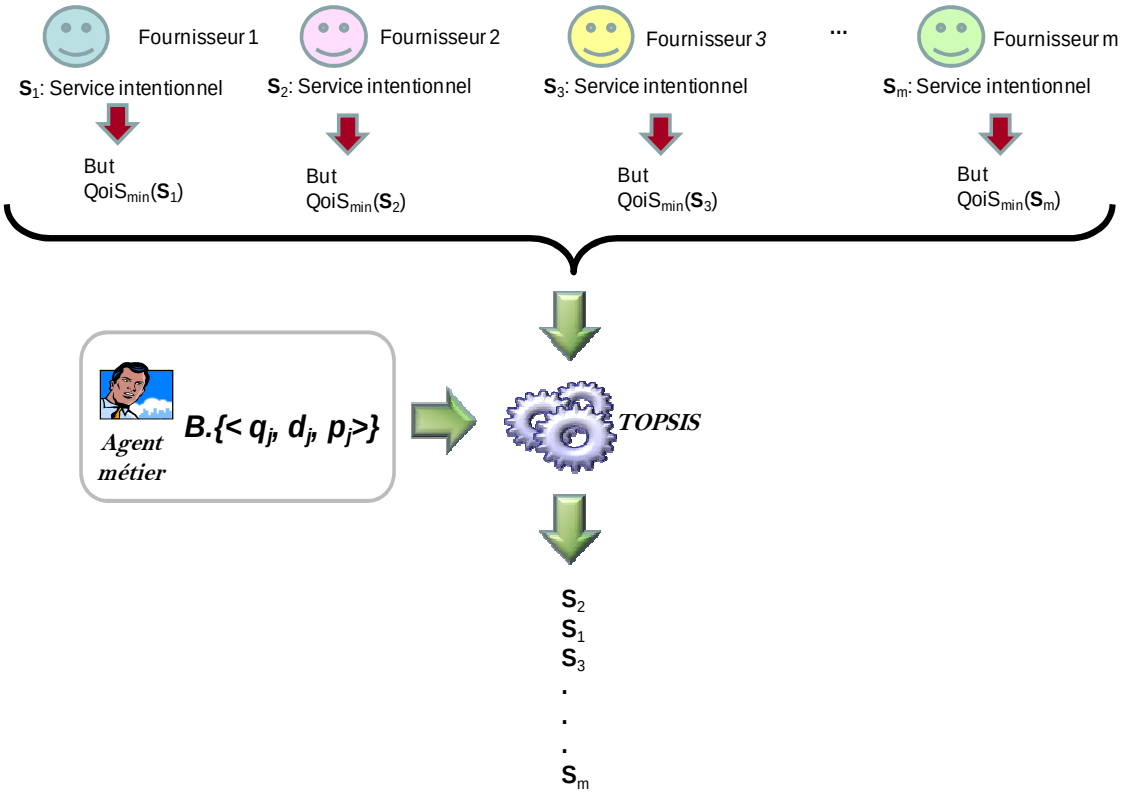

Figure 104. Exemple de sélection intentionnelle des services avec TOPSIS

La matrice de décision relative à la sélection intentionnelle des services, telle que définie à la Figure 105, est composée des éléments suivants :

- L'ensemble des services intentionnels potentiels, *i.e.* ceux dont les buts (*But* dans Figure 104) sont sémantiquement équivalents au but que l'agent métier souhaite

réaliser (*B* dans Figure 104) : $S= \{S_1, S_2, \ldots, S_i, \ldots, S_m\}$, tel que *m* est le nombre des services ($\mathbf{S}_{i/i=1..m}$) ;

- L'ensemble des buts qualité que l'agent métier désire satisfaire : $Q= \{q_1, q_2, \ldots, q_j, \ldots, q_n\}$, tel que *n* est le nombre de buts qualité ($q_{j/j=1..n}$) ;
- L'ensemble des priorités affectées par l'agent métier aux différents buts qualité : $P= \{p_1, p_2, \ldots, p_j, \ldots, p_n\}$, tel que *n* est le nombre de buts qualité ($p_{j/j=1..n}$) ;
- L'évaluation de chaque service $\mathbf{S}_i$ correspond à la qualité de ce service (*QoiS- Quality of intentional Service*), c'est-à-dire le seuil de satisfaction de chaque service $\mathbf{S}_i$ concernant chaque but qualité q_j (*i.e.* les qualités désirées de l'agent métier). Il s'agit, particulièrement, de la QoiS_{min}, publiée pour chaque service agrégat (*cf.* Chapitre 6). Ainsi, la matrice de décision contient pour un but qualité q_j et un service S_i, le seuil minimal de satisfaction d_{ij}, publié par le service $\mathbf{S}_i$ via sa QoiS_{min}. Ce seuil de satisfaction peut être « ++ » (très satisfait), « + » (satisfait), « - » (peu satisfait) et « -- » (pas du tout satisfait). On associe un seuil égale à « *?* » (neutre) lorsque le service n'a aucune contribution à un but qualité donné.

La Figure 105 illustre la matrice de décision des services intentionnels relative à la Figure 104.

	But qualité				
Services	q_1	q_2	…	q_j …	q_n
	p_1	p_2	…	p_j …	p_n
$\mathbf{S}_1$	d_{11}	d_{12}		d_{1j}	d_{1n}
$\mathbf{S}_2$	d_{21}	d_{22}		d_{2j}	d_{2n}
…	…				
$\mathbf{S}_i$	d_{i1}	d_{i2}		d_{ij}	d_{in}
…	…				
$\mathbf{S}_m$	d_{m1}	d_{m2}		d_{mj}	d_{mn}

Figure 105. La matrice de décision des services

Etape 1 : Construire la matrice normalisée

Dans notre cas, tous les buts qualité sont exprimés par des seuils de satisfaction. Par conséquent, nous n'avons pas besoin d'appliquer la formule de normalisation (*cf.* Section 8.7). Néanmoins, la méthode TOPSIS se base sur des valeurs numériques, alors nous associons aux seuils de satisfaction des valeurs numériques (*cf.* Chapitre 6). Le Tableau 24 rappelle la valeur numérique associée à chaque seuil de satisfaction.

Tableau 24. Valeur numérique pour les seuils de satisfaction

Seuil de satisfaction	Valeur
++	+2
+	+1
?	0
-	-1
--	-2

La normalisation de la matrice de décision produit la matrice *R* et elle se fait comme suit :

$$r_{ij} = valeur(Seuil_{ij})$$

$$i=1 \ldots m;\ j=1 \ldots n$$

{(++ remplacé par 2), (+ remplacé par 1),

(? remplacé par 0), (- remplacé par -1), (-- remplacé par -2)}

Etape 2 : Construire la matrice normalisée pondérée

Soit p_j où $j=1,2,\ldots,n$ les priorités que l'agent métier associe à chacun des buts qualité qu'il souhaite satisfaire. Il s'agit alors de multiplier chaque valeur de la matrice normalisée r_{ij} par la priorité correspondant p_j. La pondération des éléments de la matrice *R* produit la matrice *V*. Les éléments de *V* se calculent comme suit :

$$v_{ij} = r_{ij} \times p_j$$

$$i=1 \ldots m;\ j=1 \ldots n$$

Etape 3 : Définir les alternatives artificielles

Il s'agit d'identifier les services correspondant aux alternatives artificielles de TOPSIS : l'alternative idéale et la pire alternative. L'objectif de l'adaptation de TOPSIS est de sélectionner les services qui se rapprochent le plus du contexte qualité de l'agent métier. C'est-à-dire, nous devons trouver les services, lesquels sont d'une part, les plus proches de ce que désire l'agent métier, et d'autre part les plus éloignés du contraire de ce qu'il désire. Nous

proposons, alors, de considérer le contexte qualité de l'agent métier, c'est à dire $\{< q_j, d_j, p_j >\}$, comme étant l'alternative idéale. Aussi, nous définissons la négation du contexte qualité de l'agent métier comme étant la pire alternative, celle qui s'éloigne le plus de ce que souhaite l'agent métier. Ainsi :

- le contexte qualité de l'agent métier est défini comme étant l'ensemble des valeurs numériques pondérées des seuils de satisfaction d_j. Il est défini comme suit :

$$\boldsymbol{A}^* = \{d^*_1, \ldots, d^*_j\} \qquad d^*_j = valeur\ (d_j) \times p_j \text{ et } d_j, p_j \in \{< q_j, d_j, p_j >\}$$

- la négation du contexte qualité de l'agent métier est définie comme étant la négation de l'ensemble des valeurs numériques pondérées des seuils de satisfaction désirés par l'agent métier. Nous proposons, comme le montre le Tableau 25**,** la négation de la valeur numérique des seuils de satisfaction, à savoir « + », « ++ », « - » et « -- ». *A'* est défini comme suit :

$$\boldsymbol{A}' = \{d'_1 \ldots d'_j\} \qquad d'_j = -1 * d^*_j$$

Tableau 25. Définition de d'_j

d^*_j	d'_j
2	-2
1	-1
0	0
-1	1
-2	2

Etape 4 : Calculer les distances euclidiennes.

Il s'agit, d'abord, de calculer la distance euclidienne S_i^* de chaque service $\boldsymbol{S}_i$, par rapport au contexte qualité de l'agent métier *A**, c'est-à-dire, de combien le service $\boldsymbol{S}_i$ est proche de *A**.

$$S_i^* = [\ \Sigma (d_j^* - v_{ij})^2\ ]^{½} \qquad i = 1, \ldots, m$$

Ensuite, il faut calculer la distance euclidienne S'_i de chaque service $\boldsymbol{S}_i$, par rapport à la négation du contexte qualité de l'agent métier *A'*, c'est-à-dire, de combien le service $\boldsymbol{S}_i$ est proche de *A'*.

$$S'_i = [\ \Sigma (d_j' - v_{ij})^2\ ]^{½} \qquad i = 1, \ldots, m$$

Etape 5 : Calculer la proximité (ou la similarité) relative.

Calculer la proximité relative C_i^* permet de comparer les différents services $\boldsymbol{S}_i$ par rapport aux alternatives artificielles *A** et *A'*. Il est ainsi considéré comme étant le degré de préférence de l'agent métier par rapport aux *m* services.

$$C_i^* = S'_i / (S_i^* + S'_i) \qquad i = 1, \ldots, m \qquad 0 < C_i^* <= 1$$

Il s'agit ici de maximiser $C^*_{i=1..m}$: les services dont les $C^*_{i=1..m}$ sont proches de *1* sont d'une part les plus proches du contexte qualité de l'agent métier, et d'autre part les plus éloignées de la négation de ce contexte qualité. Par conséquent, lorsque $C^*_{i=1..m}$= 1 alors $\boldsymbol{S}_i = A^*$, c'est-à-dire que l'on a trouvé des services, lesquels correspondent complètement au contexte qualité de l'agent métier. Si $C^*_{i=1..m} = 0$ alors $\boldsymbol{S}_i = A'$, c'est-à-dire qu'il s'agit de la négation du contexte qualité.

Étape 6 : classer les services selon le degré de préférence.

Nous classons les services sélectionnés suivant le degré de préférence, à savoir la proximité relative ($C^*_{i/i=1..m}$). Les meilleurs services seront ceux qui ont, d'une part, la plus courte distance de A*, et d'autre part, la plus longue distance de A'. Par exemple, à la Figure 104, le résultat de l'adaptation de TOPSIS donne le classement suivant : $\mathbf{S}_2 > \mathbf{S}_1 > \mathbf{S}_3 \ldots > \mathbf{S}_m$. Cela signifie que le service $\mathbf{S}_2$ est le plus proche du contexte qualité de l'agent métier. La Figure 106 résume l'algorithme de la méthode TOPSIS, adaptée à la sélection intentionnelle des services.

```
TOPSIS_SelectionIntentionnelleService ()

ENTREE
m : nombre de services
n : nombre de buts qualité
S[i] : les services intentionnels réalisant « B »
Cq = B.<qj, dj, pj>}: le contexte qualité de l'agent métier
q[j] : les buts qualité souhaités ; d[j] : les seuils de satisfaction
souhaités ; p[j] : les priorités souhaitées
d[i,j] : le seuil de satisfaction de chaque service S[i]par rapport aux
buts qualité d[j]
M : matrice de décision de (m×n) éléments
R : matrice de décision normalisée
V : matrice de décision normalisée pondérée

SORTIE
ListeServices[i] : classement descendant des services dont C* est proche
de 1

DEBUT
   M[i,j]=ConstruireM (S[i],m,q[j],n)
   R[i,j]=ConstruireR (M)
   V[i,j]=ConstruireV (R)
   A*[j]=CalculerA*({<qj, dj, pj>})
   A'[j]=CalculerA'(A*)
   S*[i]=CalculerS* (V[i,j],A*[j])
   S'[i]=Calculer (V[i,j],A'[j])
   C*[i]=Calculer (S*[i],S'[i])
   ListeAlternative[i]=Classer(C*[i])

   ConstruireM (S[i],m,q[j],n)
   { pour i=1,m
         pour j=1,n
                  d[i,j]= QoiSmin(S[i], q[j])
   }
   ConstruireR (M)
```

```
   { pour i=1,m
            pour j=1,n
                  Si d[i,j]= ++ alors r[i,j] = 2
                   Si d[i,j]= + alors r[i,j] = 1
                   Si d[i,j]= ? alors r[i,j] = 0
                   Si d[i,j]= - alors r[i,j] = -1
                   Si d[i,j]= -- alors r[i,j] = -2
   }
   ConstruireV (R)
   { pour i=1,m
            pour j=1,n
   }              v[i,j] = p[j] × r[i,j]
   CalculerA* ({< qj, dj, pj>})
   { pour j=1,n
            d*[j]= dj × pj  // dj, pj ∈ {< qj, dj, pj>}
            A*[j]= d*[j]
   }
   CalculerA' (A*)
   { pour j=1,n
            d'[j]= -1 * d*[j]   // d*[j] ∈ A*
            A'[j]= d'[j]
   }
   CalculerS* (V[i,j],A*[j])  // distance du contexte qualité
   { pour i=1,m
            pour j=1,n
               S*[i] =[ Σ (v*[j] – v[i,j])² ]½
   }
   CalculerS' (V[i,j],A'[j]) // distance de la négation du contexte
   qualité
   { pour i=1,m
            pour j=1,n
                S'[i] =[ Σ (v'[j] – v[i,j])² ]½
   }
   CalculerC* (S*[i],S'[i]) // proximité relative ou degré de préférence
   { pour i=1,m
          C*[i] = S'[i]/(S*[i] + S'[i])   // 0 <  C*[i] < 1
   }
   Classer (C*[i])
   { ListeService[i] = les services classés dans un ordre descendant
    par rapport à C*[i] }
FIN
```

Figure 106. Algorithme de l'adaptation de TOPSIS dans la sélection intentionnelle des services

La sélection intentionnelle des services permet d'aider l'agent métier à choisir parmi plusieurs services, classés selon son contexte qualité. Aussi, il est possible de configurer le service agrégat, sélectionné par rapport à sa qualité minimale (*i.e.* la $QoiS_{min}$), afin de retrouver les configurations de ce service qui pourraient se rapprocher davantage du contexte qualité de l'agent métier.

8.7.2 *ADAPTATION DE TOPSIS DANS LA CONFIGURATION INTENTIONNELLE DU SERVICE AGREGAT*

Comme le montre la Figure 107, l'agent métier peut vouloir configurer les services intentionnels agrégats, précédemment sélectionnés (*cf.* Section 8.7.1), afin de rechercher une meilleure qualité, que celle exposée par la $QoiS_{min}$ lors de la publication du service par le fournisseur. Par exemple, à la Figure 107, nous supposons que l'agent métier souhaite configurer le service $\mathbf{S}_2$, qui est publié par le Fournisseur 2. Ce service (*i.e.* $\mathbf{S}_2$) réalise le but « *B* » et fournit une qualité de service $QoiS_{min}(\mathbf{S}_2)$. Le service agrégat $\mathbf{S}_2$ encapsule plusieurs points de variation. Il est alors possible de configurer ce service afin de trouver les variantes les plus avantageuses pour l'agent métier.

Nous adaptons également la méthode TOPSIS pour configurer le service intentionnel agrégat $\mathbf{S}_2$ afin de trouver les configurations $\mathbf{Conf}_{2.1}$, $\mathbf{Conf}_{2.2}$, $\mathbf{Conf}_{2.3}$…$\mathbf{Conf}_{2.i/i=1..m}$ (*m* est le nombre de configurations) de ce service qui se rapprochent le plus du contexte qualité de l'agent métier.

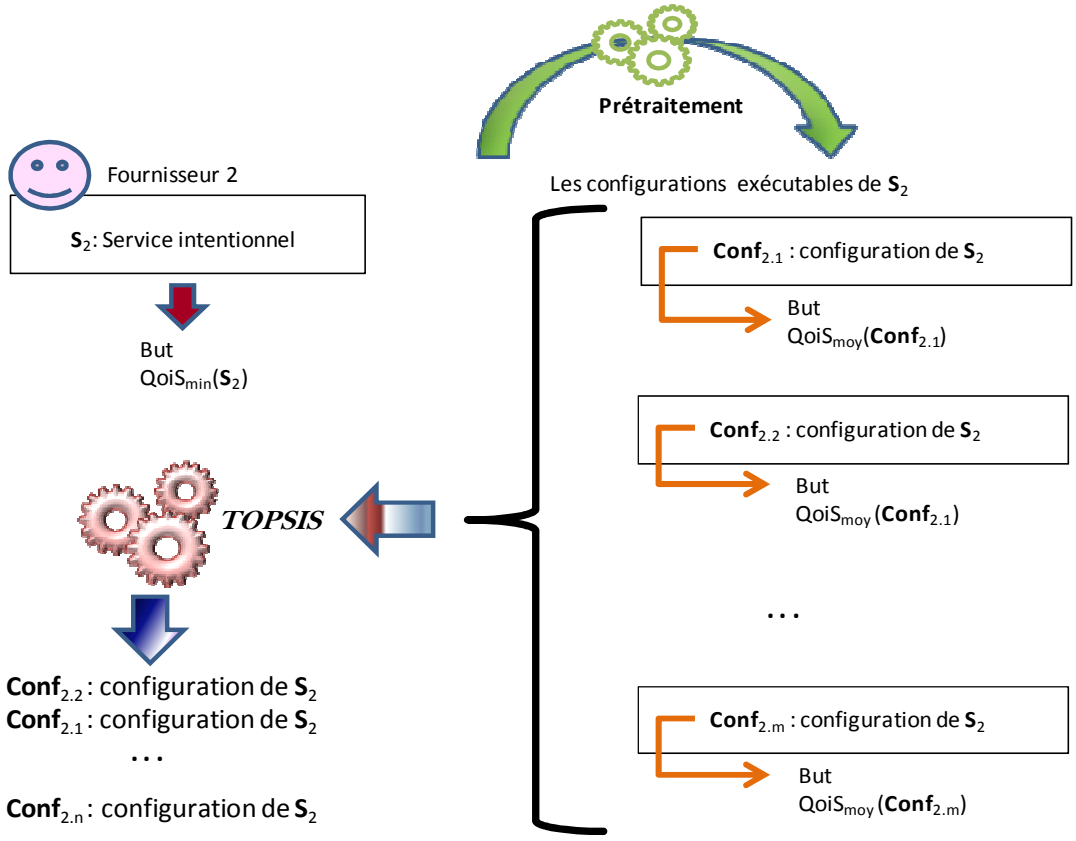

Figure 107. Exemple de configuration intentionnelle d'un service agrégat avec TOPSIS

La configuration intentionnelle du service agrégat suit le même procédé que celui de la sélection intentionnelle des services. Il s'agit de remplacer les services par les différentes configurations exécutables du service agrégat et la $QoiS_{min}$ par la $QoiS_{moy}$. Cependant, avant

de construire la matrice de décision de la Figure 108, nous devons procéder à un prétraitement (*cf.* Prétraitement à la Figure 107), lequel permet de générer les configurations, vérifier leur exécutabilité et enfin calculer leur $QoiS_{moy}$. Ainsi, dans le cas de la configuration, nous proposons que l'étape 1 corresponde à ce prétraitement. Ce dernier tient compte des éventuelles contraintes définies dans le cadre du service agrégat sélectionné, éliminant ainsi les configurations rendues non-exécutables par ces contraintes (*cf.* Chapitre 6).

Etape 1 : Prétraitement

Le service agrégat correspond à une agrégation de services composites et de services à variation (*cf.* Chapitre 4). Le prétraitement consiste à définir les différentes configurations exécutables à comparer, les buts qualité qui correspondent aux critères de sélection, et les $QoiS_{moy}$ de ces configurations.

GENERER LES CONFIGURATIONS DU SERVICE AGREGAT

La génération des différentes configurations du service agrégat correspond à un choix successif de variantes. Chaque configuration du service agrégat, désignée par le code $\boldsymbol{Conf}_{i/i=1..m}$, correspond à un service composite séquentiel (*i.e.* un chemin). La génération des configurations du service agrégat dépend de ses services composants et de la nature des opérateurs qui relient ces composants.

Dans le cas de service à choix alternatif ou de service à variation de chemin, le nombre de configurations correspond au nombre de variantes de ces services. Par exemple, à partir du service $\otimes(\mathbf{S}_1, \mathbf{S}_2, \mathbf{S}_3)$, nous pouvons générer trois configurations, $\mathbf{S}_1$, $\mathbf{S}_2$, ou $\mathbf{S}_3$, car ceux-ci sont reliés par un opérateur dont la sémantique peut être assimilée à un *Ou exclusif* : une seule option peut alors être choisie. Pour le service à choix multiple, le nombre de configurations est égal à (2^n-1), *n* étant le nombre de variantes. Par exemple, à partir du service $v(\mathbf{S}_1, \mathbf{S}_2)$, nous pouvons générer trois configurations, $\mathbf{S}_1$, $\mathbf{S}_2$, ou $\bullet(\mathbf{S}_1, \mathbf{S}_2)$, car ces variantes sont reliées par un opérateur dont la sémantique peut être assimilée à un *Ou* : soit on choisit $\mathbf{S}_1$, soit $\mathbf{S}_2$, soit les deux. Enfin, si le service composant est un service séquentiel, alors le nombre de configurations correspond au service lui-même, car ce dernier ne comprend pas de la variation.

Soit *m* le nombre de configurations, l'indice *i* indique le numéro de la configuration, relatif au numéro du service agrégat. Par exemple, à la Figure 107, nous supposons que le service agrégat $\mathbf{S}_2$ est définie par la formule suivante : $\bullet(\mathbf{S}_{ab1}, \cup(\bullet(\mathbf{S}_{bc1}, \mathbf{S}^*_{cc1}, v(\mathbf{S}_{cd1}, \mathbf{S}_{cd2}), \mathbf{S}_{de1}), \bullet(\mathbf{S}_{bd1}, \mathbf{S}_{de1})))$. Cette dernière indique que pour réaliser le but du service $\mathbf{S}_2$, il y a plusieurs services à variation à considérer, à savoir $v(\mathbf{S}_{cd1}, \mathbf{S}_{cd2})$ et $\cup(\bullet(\mathbf{S}_{bc1}, \mathbf{S}^*_{cc1}, v(\mathbf{S}_{cd1}, \mathbf{S}_{cd2}), \mathbf{S}_{de1}),$

● ($\mathbf{S}_{bd1}$, $\mathbf{S}_{de1}$)). A partir de la formule du service $\mathbf{S}_2$, différentes configurations peuvent alors être générées, en faisant des choix successifs des variantes, jusqu'à ce que chaque configuration corresponde à un service séquentiel (*cf.* Tableau 17). Par exemple, nous avons les configurations $\mathbf{Conf}_{2.1}$=● ($\mathbf{S}_{ab1}$, $\mathbf{S}_{bc1}$, $\mathbf{S}_{cd1}$, $\mathbf{S}_{de1}$), $\mathbf{Conf}_{2.2}$=● ($\mathbf{S}_{ab1}$, $\mathbf{S}_{bc1}$, $\mathbf{S}^*_{cc1}$, $\mathbf{S}_{cd1}$, $\mathbf{S}_{de1}$), *etc*.

VALIDER LES CONFIGURATIONS DU SERVICE AGREGAT

Les configurations générées doivent néanmoins être exécutables. Elles doivent satisfaire les contraintes de dépendance existantes entre les variantes du service agrégat. En effet, à chaque service agrégat peut correspondre un ensemble de contraintes de dépendance Exige et Exclut. Ces contraintes concernent les variantes de ce service et expriment les services qui ne peuvent pas faire partie d'une même configuration (*i.e.* contrainte Exclut), et ceux qui, au contraire, doivent en faire partie (*i.e.* contrainte Exige). Une configuration $\boldsymbol{Conf}_i$ est dite exécutable lorsque toutes les contraintes, associées au service agrégat, sont satisfaites (*cf.* Chapitre 6).

Pour chaque service participant à la configuration, les contraintes associées au service devront être évalues. Ainsi, lorsqu'un service source ne fait pas partie d'une configuration, le service cible n'est alors pas évalué, car la contrainte ne concerne pas cette configuration. Contrairement, si un service source est présent dans une configuration, il est impératif d'évaluer le service cible : celui-ci doit être présent dans cette configuration, s'il s'agit d'une contrainte d'exigence, et il ne doit pas en faire partie, s'il s'agit d'une contrainte d'exclusion (*cf.* Section 6.3.1.4.3). Chaque configuration est ainsi validée, en fonction des contraintes du service agrégat la concernant. Si une de ces contraintes n'est pas vérifiée, la configuration est alors considérée comme non-exécutable, et elle est exclue de la liste des configurations considérées par la suite.

CALCULER LA $\mathbf{QoiS}_{MOY}$ DES CONFIGURATIONS EXECUTABLES DU SERVICE AGREGAT

Une fois les configurations exécutables du service agrégat définies, il s'agit alors de calculer leur qualité moyenne $\mathrm{QoiS}_{\mathrm{moy}}$, et de les classer en fonction de cette valeur. La $\mathrm{QoiS}_{\mathrm{moy}}$ de chaque configuration exécutable est calculée dynamiquement, pour chaque qualité que l'agent métier souhaite satisfaire, afin d'affiner la comparaison des services (*cf.* Section 6.3.2.2.4).

La $\mathrm{QoiS}_{\mathrm{min}}$ avantage la garantie d'un seuil de satisfaction, par une agrégation intentionnelle de service, en propageant le seuil minimal. Cependant, se baser sur la qualité minimale, pour comparer les configurations de cette agrégation fournit des résultats qui privilégient les configurations ayant un « + » et plusieurs « - » par rapport à celles qui possèdent un seul « - » et plusieurs « + ». Ainsi, nous proposons de comparer les configurations d'une agrégation, non plus sur la base de la qualité minimale, mais en utilisant une qualité moyenne. Avant de

calculer la qualité moyenne, les qualités qualitatives (« + », « ++ », « - », « -- », « ? ») des services atomiques sont d'abord normalisées, comme présenté dans la sélection de service (*cf.* Tableau 24), car le calcul de la $QoiS_{moy}$ se base sur des valeurs quantitatives. Ensuite, la $QoiS_{moy}$ est calculée pour chaque configuration exécutable (*cf.* Chapitre 6). Lorsqu'une configuration ne fournit aucune des qualités que l'agent métier souhaite satisfaire, on lui affecte la valeur *0* à ces qualités. Ceci permet de considérer cette configuration dans le processus de comparaison, malgré l'absence de certaines qualités.

Afin d'adapter les différentes étapes de la méthode TOPSIS (*cf.* Section 8.7) à la configuration intentionnelle des services, il faut d'abord construire la matrice de décision correspondant aux qualités moyennes de chaque configuration disponible. Du prétraitement résulte la matrice de décision normalisée *R*, telle que illustrée à la Figure 108. Cette matrice est composée des éléments suivants :

- l'ensemble des configurations exécutables $\boldsymbol{Conf}_{i/i=1..m}$ du service intentionnel agrégat, définies lors du prétraitement. Celles-ci réalisent forcément le but de l'agent métier, mais fournissent des qualités différentes ;
- comme la sélection intentionnelle, l'ensemble des buts qualité que l'agent métier désire satisfaire, ainsi que les priorités associées à chaque but qualité: $Q= \{q_1, q_2, \ldots, q_j, \ldots, q_n\}$ et $P= \{p_1, p_2, \ldots, p_j, \ldots, p_n\}$, tel que *n* est le nombre de buts qualité (*cf.* Section 8.7.1) ;
- l'évaluation de chaque configuration $\mathbf{Conf}_i$, correspondant au seuil de satisfaction d_{ij} que chaque configuration $\mathbf{Conf}_i$ peut fournir concernant les buts qualité q_j. Il s'agit particulièrement de la $QoiS_{moy}$ calculée lors du prétraitement. Contrairement à la $QoiS_{min}$, publiée par les fournisseurs et utilisée pour comparer les services agrégats (*cf.* Section 8.7.1), la $QoiS_{moy}$ est calculée pour chaque configuration exécutable et est utilisée afin de comparer ces configurations.

Nous considérons donc que le service $\mathbf{S}_2$ possède *m* configurations ($\mathbf{Conf}_{2./i=1\ldots m}$) et *n* buts qualité ($q_{j/j=1\ldots n}$). La Figure 108 illustre la matrice de décision des configurations du service $\mathbf{S}_2$ relative à la Figure 107.

Une fois la matrice de décision *R* calculée, à partir de la qualité moyenne, l'application de la méthode TOPSIS à la configuration de service suit les mêmes étapes (étape 2 à étape 6) que celles utilisées par la sélection (*cf.* Section 8.7.1). Ces étapes appliquées au cas de la configuration sont détaillées dans ce qui suit.

	But qualité					
Configurations	q_1	q_2	…	q_j	…	q_n
	p_1	p_2	…	p_j	…	p_n

Conf$_{2.1}$	d_{11}	d_{12}	d_{1j}	d_{1n}
Conf$_{2.2}$	d_{21}	d_{22}	d_{2j}	d_{2n}
…	…			
Conf$_{2.i}$	d_{i1}	d_{i2}	d_{ij}	d_{in}
…	…			
Conf$_{2.m}$	d_{m1}	d_{m2}	d_{mj}	d_{mn}

Figure 108. La matrice de décision *R* des configurations du service $\mathbf{S_2}$

Etape 2 : Construire la matrice pondérée

La pondération des éléments de la matrice *R* aboutit à la matrice *V*, exactement de la même manière que pour la sélection intentionnelle (*cf.* Section 8.7.1).

Etape 3 : Définir les solutions artificielles

Comme dans la sélection intentionnelle (*cf.* Section 8.7.1), le contexte qualité de l'agent métier correspond à l'alternative idéale (*i.e. A**) et la négation du contexte qualité correspond à la pire alternative (*i.e. A'*).

Etape 4 : Calculer les distances euclidiennes.

Il s'agit de calculer les distances euclidiennes S_i^* et S'_i de chaque configuration $\boldsymbol{Conf}_{2.i}$, par rapport respectivement à *A** et *A'* (*cf.* Section 8.7.1).

Etape 5 : Calculer la proximité (ou la similarité) relative.

Cette étape permet de calculer la proximité relative C_i^* (*cf.* Section 8.7.1) permettant de comparer les différentes configurations $\boldsymbol{Conf}_{2.i}$ par rapport aux alternatives artificielles *A** et *A'*. Il est considéré comme étant le degré de préférence des *m* configurations.

En maximisant $C^*_{i=1..m}$, les configurations dont les $C^*_{i=1..m}$ sont proches de *1* seront d'une part les plus proches du contexte qualité, et d'autre part les plus éloignées de la négation du contexte qualité. Donc, lorsque $C^*_{i=1..m}= 1$ alors $\boldsymbol{C}_i = A^*$, c'est-à-dire que l'on a trouvé des configurations, qui sont identiques au contexte qualité de l'agent métier. Si $C^*_{i=1..m} = 0$ alors $\boldsymbol{C}_i = A'$, c'est-à-dire qu'il s'agit de la négation du contexte qualité.

Étape 6 : classer les configurations selon le degré de préférence.

Il s'agit de classer les configurations suivant le degré de préférence, à savoir $C^*_{i/i=1..m}$. Les meilleures configurations étant celles qui possèdent, d'une part, la plus courte distance de A*, et d'autre part, la plus longue distance de A'. Par exemple, à la Figure 107, le résultat de l'adaptation de TOPSIS donne le classement suivant : $\mathbf{Conf}_{2.3}$> $\mathbf{Conf}_{2.4}$ > $\mathbf{Conf}_{2.1}$…> $\mathbf{Conf}_{2.i/i=1..m}$. Cela signifie que la configuration $\mathbf{Conf}_{2.3}$ est la plus proche du contexte qualité

de l'agent métier et la plus éloignée de la négation du contexte qualité. La Figure 109 résume l'algorithme de la méthode TOPSIS, adaptée à la configuration intentionnelle des services. Cet algorithme est similaire à celui de la sélection, illustré à la Figure 106, sauf que nous remplaçons les services par les configurations et que nous rajoutons le prétraitement.

```
TOPSIS_ConfigurationIntentionnelleService ()

ENTREE
m : nombre de configurations
n : nombre de buts qualité
S[i] : le service agrégat à configurer, réalisant « B »
CD  = {cd_l} ensemble des contraintes de dépendance définies dans S[i]
Cq = B.<q_j, d_j, p_j>}: le contexte qualité de l'agent métier
q[j] : les buts qualité souhaités ; d[j] : les seuils de satisfaction
souhaités ; p[j] : les priorités souhaitées
d[i,j] : le seuil de satisfaction de chaque service S[i] par rapport aux
buts qualité q[j]
R : matrice de décision normalisée
V : matrice de décision normalisée pondérée

SORTIE
ListeConfigurations[i] : classement descendant des configurations  dont C*
est proche de 1

DEBUT
   Conf[i]= GénérerConfiguration (S[i])
   Conf[i]= ValiderConfigurationExécutable(Conf[i],D)
   R[i,j]= ConstruireR (Conf[i],m,q[j],n)
   V[i,j]= ConstruireV (R)
   A*[j]= CalculerA*({<q_j, d_j, p_j>})
   A'[j]= CalculerA'(A*)
   S*[i]= CalculerS* (V[i,j],A*[j])
   S'[i]= Calculer (V[i,j],A'[j])
   C*[i]= Calculer (S*[i],S'[i])
   ListeAlternative[i]= Classer(C*[i])

   GénérerConfiguration (S[i]) // génération des configurations
   {  si (service à choix alternatif(S[i])) v (service à variation de
            chemin(S[i])) alors m=i
      si (service à choix multiple(S[i]))
          alors m=(2^i-1)
      si (service séquentiel(S[i]))
          alors m=1
   }
   ValiderConfigurationExécutable (Conf[i],CD) //garder les Conf valides
   {  Pour chaque cd_l ∈ CD
        Si Conf[i] ne satisfait pas cd_l
          alors éliminer(Conf[i])
   }
   calculerR (Conf[i],m,q[j],n)
   { Si d[i,j]= ++ alors r[i,j] =2
     Si d[i,j]= + alors r[i,j] = 1
     Si d[i,j]= ? alors r[i,j] = 0
     Si d[i,j]= - alors r[i,j] = -1
     Si d[i,j]= -- alors r[i,j] = -2

     pour i=1,m
         pour j=1,n
```

```
                d[i,j]= QoiSmoy(Conf[i]/q[j])
   }
   ConstruireV (R)
   { pour i=1,m
            pour j=1,n
                 v[i,j] = p[j] × r[i,j]
   }
   CalculerA* ({< q_j, d_j, p_j>})
   { pour j=1,n
            d*[j]= d_j × p_j  // d_j, p_j ∈ {< q_j, d_j, p_j>}
            A*[j]= d*[j]
   }
   CalculerA' (A*)
   { pour j=1,n
            d'[j]= -1 * d*[j]   // d*[j] ∈ A*
            A'[j]= d'[j]
   }
   CalculerS* (V[i,j],A*[j])  // distance du contexte qualité
   { pour i=1,m
            pour j=1,n
               S*[i] =[ Σ (v*[j] – v[i,j])² ]½
   }
   CalculerS' (V[i,j],A'[j]) // distance de la négation du contexte
   qualité
   { pour i=1,m
            pour j=1,n
                S'[i] =[ Σ (v'[j] – v[i,j])² ]½
   }
   CalculerC* (S*[i],S'[i]) // proximité relative ou degré de préférence
   { pour i=1,m
          C*[i] = S'[i]/(S*[i] + S'[i])   // 0 <  C*[i] < 1
   }
   Classer (C*[i])
   {  ListeConfiguration[i] = les configurations classées dans un
      ordre descendant par rapport à C*[i]
   }

FIN
```

Figure 109. Algorithme de l'adaptation de TOPSIS dans la configuration intentionnelle du service agrégat

8.8 *CONCLUSION*

Nous avons présenté dans ce chapitre une méthodologie de sélection et de configuration intentionnelle permettant de guider l'exécution intentionnelle des services, de façon à choisir les meilleurs services, en fonction des exigences non-fonctionnelles des agents métier.

Nous avons proposé de formaliser les exigences des agents métier par un modèle de contexte qualité, afin de faciliter le processus de sélection. Le contexte qualité correspond au but que l'agent métier désire réaliser, ainsi que l'ensemble des qualités qu'il souhaite satisfaire. Dans le cadre de l'architecture iSOA, nous proposons que l'agent métier se base sur le référentiel qualité (*cf.* Chapitre 5) du domaine qui le concerne afin de décider des qualités

qu'il souhaite satisfaire. Le référentiel qualité lui permet également de vérifier la sémantique des seuils de satisfaction, afin qu'il définisse ceux qui lui conviennent.

Ensuite, nous avons présenté le processus de sélection intentionnelle des services, qui permet de classer, par degré de préférence, un ensemble de services intentionnels agrégats appartenant à plusieurs fournisseurs : des plus proches du contexte qualité de l'agent métier, aux plus éloignés. Les services agrégats sont triés selon leur $QoiS_{min}$. Le processus de configuration intentionnelle, quant à lui, permet d'abord de générer les différentes configurations du service agrégat et de valider leur exécutabilité, en vérifiant la satisfaction des contraintes de dépendance définies entre les variantes de ce service. Ensuite, ces configurations sont classées, selon leur $QoiS_{moy}$, en fonction du contexte qualité de l'agent métier.

Enfin, nous avons choisi d'appliquer les méthodes d'aide à la décision multicritère dans la sélection et la configuration des services, afin d'aider l'agent métier à trouver, en fonction des de ses préférences, un compromis entre plusieurs services offrant des qualités souvent conflictuelles. Particulièrement, nous avons adapté la méthode TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) (Hwang&Yoon,1981) pour faire la sélection et la configuration intentionnelle des services. Nous proposons, dans le Chapitre 9, le prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*) correspondant à l'implémentation de la méthode TOPSIS, adaptée à la sélection et à la configuration intentionnelle des services.

CHAPITRE 9 : PROTOTYPE TOPSIQC

9.1 *INTRODUCTION*

Ce chapitre présente le prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*), qui est développé pour supporter la méthodologie de sélection et de configuration intentionnelle des services, proposée dans le Chapitre 8. Le but de ce prototype est de montrer la faisabilité de la sélection et de la configuration intentionnelle des services que nous avons proposés dans cette thèse, en implémentant la méthode TOPSIS. TOPSiQC permet de fournir un classement des services intentionnels (ou des configurations), selon le contexte qualité de l'agent métier, c'est-à-dire les services (ou les configurations) qui se rapprochent au mieux du contexte qualité de l'agent métier.

Ce chapitre est organisé selon plusieurs sections. La section 2 présente la vue globale du prototype, tandis que la section 3 propose une vue détaillée de celui-ci. La section 4 conclut ce chapitre.

9.2 *VUE GLOBALE DU PROTOTYPE TOPSIQC*

Le prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*) est développé pour implémenter l'adaptation de la méthode TOPSIS dans le cas de la sélection et de la configuration intentionnelle des services. L'objectif de TOPSiQC est de montrer la faisabilité de la solution que nous proposons. La Figure 110 présente une vue globale du fonctionnement du prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*).

Comme le montre la Figure 110, le prototype TOPSiQC prend en entrée le contexte qualité de l'agent métier, ainsi que l'ensemble des services intentionnels à sélectionner ou le service agrégat à configurer. Dans le cas de la configuration du service agrégat, il faut considérer également les contraintes de dépendance entre les services. Celles-ci permettent de valider l'exécutabilité des différentes configurations générées (*cf.* Chapitre 6). Après un certain nombre de traitements, le prototype TOPSiQC fournit le tri de ces services intentionnels ou des configurations du service agrégat, en fonction des exigences non-fonctionnelles de l'agent métier, définies dans le contexte qualité.

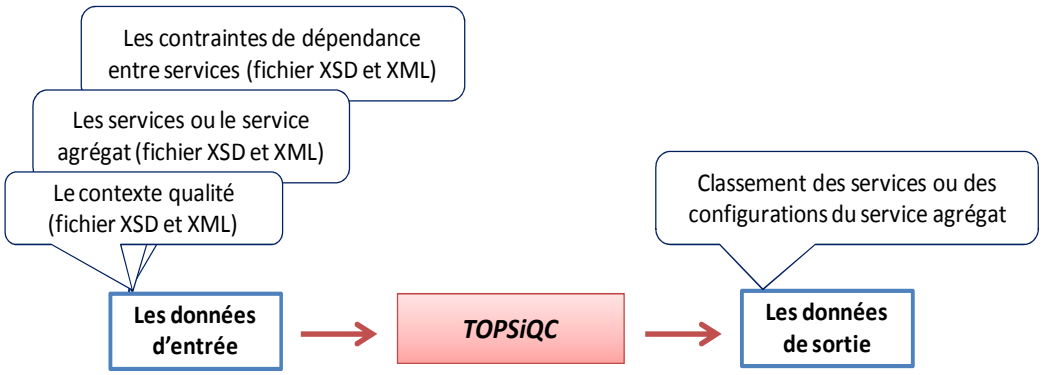

Figure 110. Vue globale du prototype TOPSiQC

Le prototype TOPSiQC est développé en *Java*. Il peut être utilisé par l'annuaire des services intentionnels, à la fois pour sélectionner parmi une multitude de services et pour configurer le service agrégat, et donc choisir parmi plusieurs configurations.

La Figure 111 présente l'architecture du prototype TOPSiQC. Celui-ci a été développé suivant l'architecture MVC (Modèle-Vue-Contrôle) (Gamma et *al.*,1997). La classe *Model* gère la logique applicative de TOPSiQC, à savoir la prise en compte de la description du service, de celle des contraintes de dépendance, de celle du contexte, ainsi que la construction et l'implémentation des différents calculs (les distances, la proximité relative…). La classe *View* s'occupe de l'IHM, tandis que la classe *Controler* s'intéresse à la gestion des évènements et à la validation des fichiers XML (contexte, contrainte et service).

Nous détaillons, dans la suite de ce chapitre, les différentes parties de l'architecture du prototype TOPSiQC, à savoir les données d'entrée (contexte, contrainte et service), les traitements, ainsi que les résultats obtenus. Nous proposons d'illustrer ces différentes parties par l'exemple de la réservation de billet de train (*cf.* Chapitre 6).

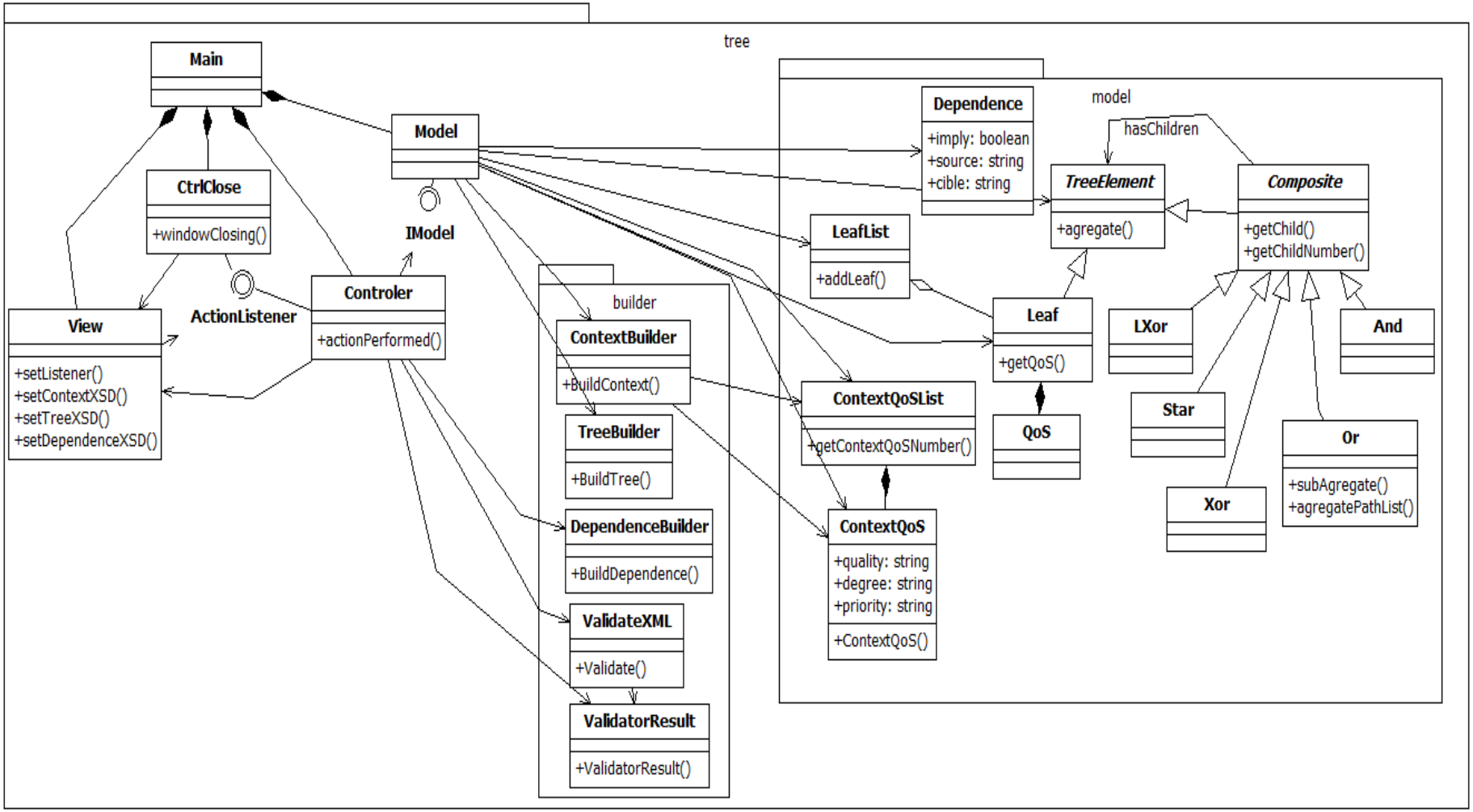

Figure 111. L'architecture du prototype TOPSiQC

9.3 *VUE DETAILLEE DU PROTOTYPE TOPSIQC*

Nous proposons, dans cette section, de détailler les différentes parties composant le prototype, à savoir les données d'entrée, les différents traitements manipulant les données d'entrée, et enfin les résultats fourni par le prototype TOPSiQC.

9.3.1 LES DONNEES D'ENTREE DE TOPSIQC

La Figure 112 présente l'interface utilisateur du prototype TOPSiQC. Elle propose via les boutons *Browse* de choisir le contexte qualité de l'agent métier, les services à sélectionner ou le service agrégat à configurer, et les contraintes de dépendance associées à ce service agrégat. Pour montrer la faisabilité de la méthode, nous avons décidé de représenter le contexte qualité, les services et les contraintes de dépendance par des fichiers XML.

Figure 112. L'IHM du prototype TOPSiQC

Afin que la description du contexte qualité, celle des services et de leurs contraintes soient conformes, la définition d'un XSD est nécessaire. Un XSD (*XML Schema Definition*) est recommandé, par le W3C en mai 2001, comme langage de description de format de document XML[17]. Il permet de définir la structure d'un document XML. La connaissance de la structure d'un document XML permet particulièrement de vérifier sa conformité. Ainsi, les définitions XSD, relatives au contexte qualité, aux services et leurs contraintes, permettent d'éclaircir la structure des documents XML concernant, respectivement, le contexte qualité, les services et leurs contraintes de dépendance. Les documents XML, relatifs au contexte qualité, aux services et leurs contraintes, représentent des instances des définitions XSD. Les boutons *Browse* permettent juste de choisir le fichier XML du contexte qualité, celui du service et celui des contraintes de dépendance. Les fichiers XSD sont utilisés par la classe *Controler* pour vérifier la conformité des schémas XML sélectionnés par les utilisateurs.

[17] http://www.w3.org/XML/Schema

9.3.1.1 *LA SPECIFICATION DU CONTEXTE QUALITE*

La spécification du contexte qualité de l'agent métier se base sur le modèle du contexte qualité présenté au Chapitre 8. Il est décrit par un XSD et l'instanciation de ce dernier par un schéma XML. La définition XSD du contexte qualité de l'agent métier est modélisée à la Figure 113, en utilisant le langage UML (*Unified Modelling Language*). L'élément *ContextQoSList* regroupe l'ensemble des qualités (*i.e. ContextQoS* de la Figure 113) que l'agent souhaite satisfaire. L'élément *ContextQoS* est défini par les attributs *quality* (correspond au but qualité à satisfaire), *degree* (correspond au seuil de satisfaction) et *priority* (correspond à la priorité octroyée à chaque but qualité) (*cf.* Section 8.3). La construction du contexte qualité est effectuée par la classe *ContextBuilder* à la Figure 111.

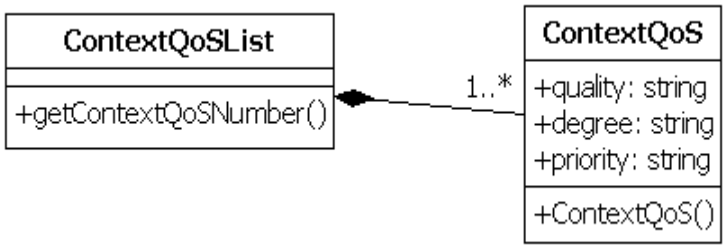

Figure 113. Représentation objet du schéma XML du contexte qualité

Un exemple de document XML du contexte qualité est présenté à la Figure 114. Par exemple, l'agent métier souhaite satisfaire la Commodité (ligne 8 à la Figure 114) et le Coût (ligne 13 à la Figure 114) avec le seuil de satisfaction « + » (lignes 9 et 14 à la Figure 114), et la même priorité « *0,5* » (lignes 10 et 15 de la Figure 114).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <context:Context xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns:context='http://Assia/Context'
5        xsi:schemaLocation='http://Assia/Context Context.xsd'>
6
7  <context:QoS>
8      <context:Quality>Commodité</context:Quality>
9      <context:Degree>+</context:Degree>
10     <context:Priority>0.5</context:Priority>
11 </context:QoS>
12 <context:QoS>
13     <context:Quality>Coût</context:Quality>
14     <context:Degree>+</context:Degree>
15     <context:Priority>0.5</context:Priority>
16 </context:QoS>
17
18 </context:Context>
19
```

Figure 114. Un exemple de contexte qualité

9.3.1.2 *LA SPECIFICATION DU SERVICE INTENTIONNEL*

La spécification du service intentionnel se base sur le modèle *MiS-q* présenté plus haut (*cf.* Chapitre 6). Il est décrit par un XSD ainsi que l'instanciation de ce dernier par un document XML. La définition XSD du service intentionnel est modélisée à la Figure 115, en utilisant le langage UML (*Unified Modelling Language*). Le service intentionnel est représenté sous forme d'un arbre (*TreeElement* à la Figure 115). Le nœud racine de l'arbre peut être un service atomique (*Leaf* à la Figure 115) ou un service agrégat (*Composite* à la Figure 115). Ce dernier peut être un service séquentiel (*And* à la Figure 115) ; un service à variation de chemin (*Xor* à la Figure 115) ; un service à choix multiple (*Or* à la Figure 115) *;* un service à choix alternatif (*LXor* à la Figure 115) ; ou un service itératif (*Star* à la Figure 115). Un service atomique (*i.e.* l'élément *Leaf*) peut fournir des qualités correspondant à l'élément *QoS* à la Figure 115. L'élément *QoS* est décrit par les attributs *quality* (*i.e.* le but qualité fourni par le service) et *degree* (*i.e.* le seuil de satisfaction fourni par le service). La construction du service est effectuée par la classe *TreeBuilder* à la Figure 111.

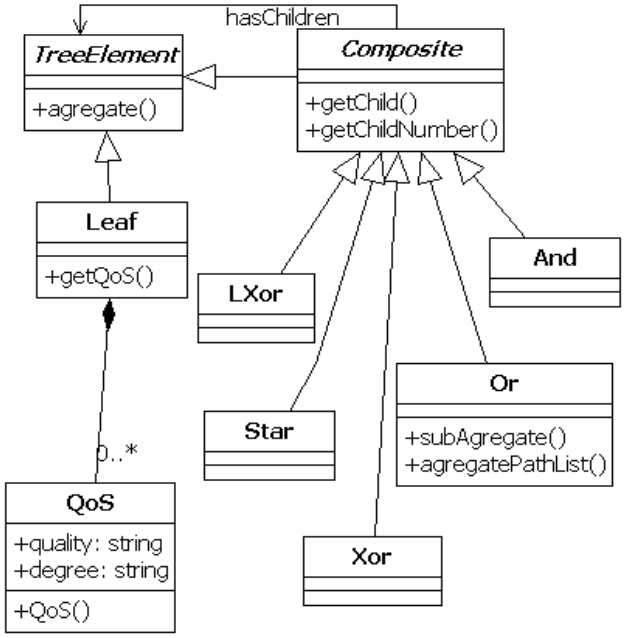

Figure 115. Représentation objet du schéma XSD du service intentionnel

Afin d'illustrer l'usage du schéma XSD que nous avons défini dans le Chapitre 6, reprenons l'exemple de la réservation de billet de train. Dans celui-ci, nous définissons le service agrégat $\mathbf{S}_{\text{Réserver un billet de train}}$ comme suit : $\mathbf{S}_{\text{Réserver un billet de train}} = \bullet(\otimes(\mathbf{S}_{\text{Réserver un billet de train par internet}}, \mathbf{S}_{\text{Réserver un billet de train en agence}}), \mathbf{S}^*_{\text{Réserver un billet de train réduit}}, \cup(\mathbf{S}_{\text{Annuler un billet de train}}, \bullet(\otimes(\mathbf{S}_{\text{Payer billet par carte bancaire}}, \mathbf{S}_{\text{Payer billet par chèque}}, \mathbf{S}_{\text{Payer billet par espèce}}), \nu(\mathbf{S}_{\text{Présenter une carte de réduction}}, \mathbf{S}_{\text{Présenter}}$

billet de train))). Ce service précise que la réservation de billet de train correspond à une composition de service (*i.e.* « ● »), qui contient plusieurs services à variation. Il s'agit d'abord de faire la réservation du billet de train (*i.e.* ●(⊗(**S**Réserver un billet de train par internet, **S**Réserver un billet de train en agence), **S***Réserver un billet de train réduit)), ensuite, effectuer le paiement du billet (*i.e.* ⊗(**S**Payer billet par carte bancaire, **S**Payer billet par chèque, **S**Payer billet par espèce)), et enfin présenter son billet le jour du voyage (*i.e.* *v*(**S**Présenter une carte de réduction, **S**Présenter billet de train)). La Figure 116 présente une description succincte de **S**Réserver un billet de train sous forme d'un arbre, sachant que les opérateurs d'agrégation « ● », « ∪», « *v* », «⊗ », et « * » sont remplacés, respectivement, par *And*, *Xor*, *Or*, *LXor* et *Star*. En effet, comme présenté plus haut (*cf.* Chapitre 4 et Chapitre 6), le service séquentiel « ● » correspond à un lien *Et* (*And* à la Figure 116) entre ses services composants ; le service à choix alternatif «⊗ », ainsi que le service à variation de chemin correspondent à un lien *Ou exclusif* (*LXor* et *Xor* respectivement à la Figure 116) entre leurs variantes ; et le service à choix multiple « *v* » correspond à un lien *Ou* (*Or* à la Figure 116) entre ses variantes. Par exemple, effectuer une réservation par Internet, via le service **S**Réserver un billet de train par internet (ligne 11 de la Figure 116), est plus commode « + » (ligne 14 à la Figure 116) que de se déplacer en agence, via le service **S**Réserver un billet de train en agence (ligne 17 à la Figure 116). La description complète de **S**Réserver un billet de train est présentée à l'Annexe 3.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2
3   <tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4         xmlns:tree='http://ApprovisionnementProduit/Tree'
5         xsi:schemaLocation='http://ApprovisionnementProduit/Tree
6   Tree.xsd'>
7
8   <tree:And id="01">
9
10    <tree:LXor id="lxor-01">
11         <tree:Leaf id="SRéserver un billet de train par internet">
12                        <tree:QoS>
13                           <tree:Quality>commodité</tree:Quality>
14                           <tree:Degree>+</tree:Degree>
15                        </tree:QoS>
16                     </tree:Leaf>
17         <tree:Leaf id="SRéserver un billet de train en agence">
18                        <tree:QoS>
19                           <tree:Quality>commodité</tree:Quality>
20                            <tree:Degree>-</tree:Degree>
21                        </tree:QoS>
22                     </tree:Leaf>
23    </tree:LXor> .....
```

Figure 116. Description du service **S**Réserver un billet de train

9.3.1.3 *LA SPECIFICATION DES CONTRAINTES DE DEPENDANCE ENTRE LES SERVICES*

La spécification des contraintes de dépendance entre les services se base sur la description des contraintes de dépendance du modèle *MiS-q* (*cf.* Chapitre 6). Il est décrit par un XSD et l'instanciation de ce dernier par un schéma XML. La définition XSD des contraintes de dépendance entre les services est modélisée à la Figure 117, en utilisant le langage UML. L'élément *Dependence* correspond aux contraintes de dépendance. Il est défini par l'attribut *imply* qui correspond au type de la contrainte. Celui-ci peut être *imply* (*i.e.* Exige) ou *exclude* (*i.e.* Exclut). La contrainte est définie par les attributs *source* et *target*, qui correspondent respectivement aux services sources et au service cible de la contrainte. Enfin, les contraintes sont valides dans le contexte d'un service agrégat donné, tel que présenté à la Figure 117. La construction des contraintes de dépendance est effectuée par la classe *DependanceBuilder* à la Figure 111.

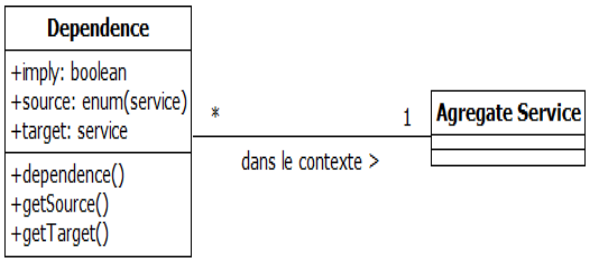

Figure 117. Représentation objet du schéma XSD des contraintes de dépendance entre services

Par exemple, les contraintes de dépendance du service agrégat $\mathbf{S}_{\text{Réserver un billet de train}}$ sont décrites à la Figure 118. En effet, le service $\mathbf{S}_{\text{Réserver un billet de train par internet}}$ exclut les services $\mathbf{S}_{\text{Payer un billet de train par espèce}}$ (ligne 8 à la Figure 118) et $\mathbf{S}_{\text{Payer un billet de train par chèque}}$ (ligne 10 à la Figure 118). D'autre part, le service $\mathbf{S}_{\text{Réserver un billet de train réduit}}$ exige le service $\mathbf{S}_{\text{Présenter une carte de réduction}}$ (ligne 12 à la Figure 118).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <dependence:Dependence xmlns:xsi='http://www.w3.org/2001/XMLSchema-
4  instance'
5        xmlns:dependence=' http://Assia/Dependence'
6        xsi:schemaLocation='http://Assia/Dependence Dependence.xsd'>
7
8  <aggregate service> Ref="S Réserver un billet de train" </aggregate service>
9
10 <dependence:Exclude source="S Réserver un billet de train par internet" target="S Payer un
11 billet de train par espèce"/>
12 <dependence:Exclude source="S Réserver un billet de train par internet" target="S Payer un
13 billet de train par chèque"/>
14 <dependence:Imply source="S Réserver un billet de train réduit" target=" S Présenter une
15 carte de réduction"/>
   </dependence:Dependence>
```

Figure 118. Les contraintes de dépendance du service $\mathbf{S}_{\text{Réserver un billet de train}}$

9.3.2 LES TRAITEMENTS DE TOPSIQC

Le déclanchement de l'ensemble des traitements qui s'opèrent sur les données d'entrées (contexte, services et contraintes de dépendance), correspondant aux différentes étapes du Chapitre 8, se fait via le bouton *Calculate Paths* de la Figure 112. Ces traitements correspondent à l'implémentation de l'adaptation de la méthode TOPSIS à la sélection et à la configuration intentionnelle des services (*cf.* Chapitre 8). Nous considérons l'exemple de la réservation de billet de train pour illustrer les différents traitements du prototype TOPSiQC.

9.3.2.1 *SELECTION INTENTIONNELLE DES SERVICES*

Nous avons présenté plus haut (*cf.* Chapitre 8) l'adaptation de la méthode TOPSIS au cas de la sélection intentionnelle des services, en fonction du contexte qualité de l'agent métier. Les contraintes de dépendance ne sont pas exploitées dans le cas de la sélection des services, car celles-ci concernent la configuration d'un service agrégat, une fois celui-ci sélectionné.

Nous supposons que l'annuaire des services intentionnels dispose de trois services intentionnels (pouvant être atomiques ou agrégats) $\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$, lesquels réalisent le but *Réserver un billet de train*. Chacun de ces services est publié par un fournisseur particulier. Nous adoptons la définition XSD du service intentionnel (*cf.* Figure 115) pour simuler la description des services intentionnels se trouvant dans l'annuaire. L'annuaire lui-même est considéré comme un *Xor* entre les multiples services qui y sont publiés. En effet, la sémantique des éléments *Xor* et *Leaf* permettent de représenter les services de l'annuaire : *Xor* précise un choix exclusif entre les services agrégats et *Leaf* précise la valeur des services (*i.e.* $\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$) et leurs qualités. La Figure 119 présente la description des services $\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$, ainsi que les qualités qu'ils fournissent.

```
<?xml version="1.0" encoding="UTF-8"?>
<tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
      xmlns:tree='http://DomaineAppProduit/Tree'
xsi:schemaLocation='http://DomaineAppProduit/Tree Tree.xsd'>

  <tree:Xor id="xor-01">
                    <tree:Leaf id=”S1">
                       <tree:QoS>
                        <tree:Quality>Commodité</tree:Quality>
                         <tree:Degree>-</tree:Degree>
                       </tree:QoS>
                       <tree:QoS>
                            <tree:Quality>Coût</tree:Quality>
                         <tree:Degree>+</tree:Degree>
                       </tree:QoS>
                    </tree:Leaf>
                     <<tree:Leaf id=”S2">
                       <tree:QoS>
                        <tree:Quality>Commodité</tree:Quality>
                         <tree:Degree>-</tree:Degree>
                       </tree:QoS>
                       <tree:QoS>
                        <tree:Quality>Coût</tree:Quality>
                          <tree:Degree>++</tree:Degree>
                        </tree:QoS>
                    </tree:Leaf>
                    <tree:Leaf id=”S3">
                       <tree:QoS>
                        <tree:Quality>Commodité</tree:Quality>
                         <tree:Degree>-</tree:Degree>
                       </tree:QoS>
                       <tree:QoS>
                        <tree:Quality>Coût</tree:Quality>
                          <tree:Degree>-</tree:Degree>
                        </tree:QoS>
                    </tree:Leaf>
  </tree:Xor>
</tree:Services>
```

Figure 119. Le document XML des services $\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$

Nous supposons également que l’agent métier souhaite satisfaire les qualités de Commodité et de Coût, telles que présentées dans le contexte qualité de la Figure 114. Il s’agit donc de guider l’agent métier à choisir parmi les services $\mathbf{S}_1$, $\mathbf{S}_2$ ou $\mathbf{S}_3$, ceux qui se rapprochent le plus de ce qu’il désire satisfaire, à savoir la Commodité et le Coût avec un seuil de satisfaction égale à « + » pour chacune des qualités. L’agent métier associe la même priorité aux deux qualités (p*riority=0,5*) (*cf.* Figure 114).

Si le fournisseur publient des services qui réalisent le but *Réserver un billet de train*, mais qui ne fournissent aucune des qualités que l’agent métier souhaite satisfaire, le prototype TOPSiQC affecte alors le seuil « ? » à ces qualités. Ceci permet alors d’intégrer ces services dans le processus de sélection, car un service qui n’offre rien (*i.e. ?*) peut être meilleur que celui qui offre des qualités négatives.

Comme présenté au Chapitre 8, une matrice de décision des services intentionnels $\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$ doit être calculée. Celle-ci contient les services à classer ainsi que les qualités qu'ils offrent. Cette matrice est implémentée par le prototype TOPSiQC (la classe *TreeModel* à la Figure 111) comme un vecteur dynamique, dit *pathList*, lequel est construit à partir de la description des services présentée à la Figure 119 (la classe *TreeBuilder* à la Figure 111). Dans le cas de la sélection intentionnelle, ce vecteur est facile à construire. En effet, il suffit de récupérer le nom des services (*i.e. id*) ainsi que leurs qualités (*i.e. QoS*) à partir du fichier de la Figure 119. Ensuite, les différentes étapes (étape 1 à étape 6) de la sélection sont implémentées par le prototype TOPSiQC. Nous détaillons dans ce qui suit chacune de ces étapes, en mettant en exergue le code permettant de les implémenter dans TOPSiQC.

Etape 1 : Construire la matrice normalisée

La Figure 120 met en évidence le code permettant de normaliser les seuils de satisfaction (étape normalisation des seuils de satisfaction à la Section 8.7.1), en affectant à ces seuils des valeurs numériques, à savoir remplacer « ++ » par *2* (ligne 3 à la Figure 120); remplacer « + » par *1* (ligne 4 à la Figure 120)*;* remplacer « *?* » par *0* (ligne 7 à la Figure 120)*;* remplacer « - » par *-1* (ligne 5 à la Figure 120)*; et* remplacer « -- » par *-2* (ligne 6 à la Figure 120).

```
1       //Storing degree values
2             HashMap degreeValue = new HashMap();
3               degreeValue.put("++", 2);
4               degreeValue.put("+", 1);
5               degreeValue.put("-", -1);
6               degreeValue.put("--", -2);
7               degreeValue.put("undefined", 0);
```

Figure 120. La normalisation des seuils de satisfaction par TOPSiQC

Etape 2 : Construire la matrice normalisée pondérée

Cette étape permet de prendre en compte les préférences de l'agent métier, en termes de priorités associées à chaque qualité. La Figure 121 (particulièrement, ce qui est en gras) montre TOPSiQC qui récupère d'une part les valeurs des qualités (ligne 2 à la Figure 121) et les priorités (ligne 4 à la Figure 121), et d'autre part pondère les valeurs des qualités des services (*i.e. priority*Vij* à la Figure 121).

```
1   //Vij
2        int Vij = ((Integer) qosNameList.get(qosName)).intValue();
3   //priority
4        double priority = (new Double(cQoS.getPriority())).doubleValue();
5    //sum(Qj*-Vij)²
6        sumSStar = sumSStar + (Math.pow((priority*Qj - priority*Vij),2));
7   //sum(Qj'-Vij)²
8        sumSPrim = sumSPrim + (Math.pow((-Qj*priority - priority*Vij),2));
```

Figure 121. La pondération dans TOPSiQC

Etape 3 : Définir les alternatives artificielles

L'alternative idéale *A** correspond au contexte qualité (*i.e. « Qj »* à la Figure 122) de l'agent métier, tandis que la pire alternative *A'* correspond à la négation du contexte qualité de l'agent métier (*i.e. « -Qj »* à la Figure 122). La Figure 122 (particulièrement, ce qui est en gras) montre le code de TOPSiQC permettant de récupérer, à partir des valeurs du contexte qualité, *A** et *A'* (*i.e.* respectivement *« priority*Qj» et « –Qj*priority »* à la Figure 122).

```
//For each quality defined in the context.
   Set keySet = qosNameList.keySet();
   Iterator qosNameListIterator = keySet.iterator();
   while(qosNameListIterator.hasNext())
…   //Qj
    int Qj = ((Integer)degreeValue.get(cQoS.getDegree())).intValue();
                //priority
       double priority = (new Double(cQoS.getPriority())).doubleValue();
                //sum(Qj*-Vij)²
       sumSStar = sumSStar + (Math.pow((priority*Qj - priority*Vij),2));
                //sum(Qj'-Vij)²
       sumSPrim = sumSPrim + (Math.pow((-Qj*priority - priority*Vij),2));
     }
```

Figure 122. Récupération du contexte qualité par TOPSiQC

Etape 4 : Calculer les distances euclidiennes

La Figure 123 illustre comment le prototype TOPSiQC calcule les distances euclidiennes S^* et S' (*i.e.* respectivement *« sumSStar »* et *« sumSPrim »* à la Figure 123). Celles-ci précisent la distance de chaque service par rapport à, respectivement, *A** et *A'*.

```
//sum(Qj*-Vij)²
     sumSStar = sumSStar + (Math.pow((priority*Qj - priority*Vij),2));
//sum(Qj'-Vij)²
     sumSPrim = sumSPrim + (Math.pow((-Qj*priority - priority*Vij),2));
//S*
     double resSStar = Math.sqrt(sumSStar);
//S'
     double resSPrim = Math.sqrt(sumSPrim);
```

Figure 123. Le calcul de S^* et S' par TOPSiQC

Etape 5 : Calculer la proximité (ou la similarité) relative.

Tel que défini dans le Chapitre 8, le calcul de la proximité relative C^* permet de situer les services ayant la plus courte distance de *A** et la plus longue distance de *A'*. La Figure 124 montre le calcul de C* par le prototype TOPSiQC.

```
//C*
     double resCStar = resSPrim / (resSPrim+resSStar);
```

Figure 124. Le calcul de C^* par TOPSiQC

Étape 6 : classer les services selon le degré de préférence

Enfin, il s'agit de classer les services en fonction du degré de préférence *C**. C'est-à-dire, ceux qui d'une part possèdent la plus courte distance du contexte qualité de l'agent métier (*i.e. A**), et d'autre part ils ont la plus longue distance de la négation de ce contexte qualité (*i.e. A'*). La Figure 125 montre les résultats obtenus par le prototype TOPSiQC concernant S*, S' et C^*. On peut s'apercevoir que le prototype TOPSiQC effectue un classement descendant des services selon C*.

Service (ou Variante)	S*	S'	C*
S2	1.5	2.0615528128088...	0.578835390393...
S1	1.581138830084...	1.5811388300841...	0.5
S3	1.581138830084...	1.5811388300841...	0.5

Figure 125. Les résultats de TOPSiQC dans le cas de la sélection des services $\mathbf{S}_1$, $\mathbf{S}_2$ et S_3

Dans le cas d'un service agrégat qui contient plusieurs variations, nous avons la possibilité de configurer ce service afin de rechercher parmi ses configurations, celles qui pourraient se rapprocher davantage du contexte qualité de l'agent métier. Ainsi, nous proposons, dans la prochaine section, de configurer le service $\mathbf{S}_2$.

9.3.2.2 *CONFIGURATION INTENTIONNELLE DES SERVICES*

La configuration intentionnelle des services permet d'affiner la sélection intentionnelle (*cf.* Section 8.7.1). Le prototype TOPSiQC implémente également la configuration intentionnelle des services. Les contraintes de dépendance sont prises en compte dans le cas de la configuration des services, afin de valider l'exécutabilité des configurations. Nous proposons de configurer le service $\mathbf{S}_2$ en utilisant TOPSiQC.

Comme présenté au Chapitre 8, la configuration intentionnelle du service agrégat suit le même procédé que celui de la sélection intentionnelle des services. Néanmoins, dans le cas de la configuration l'étape 1 correspond à un prétraitement, lequel permet de générer les configurations, vérifier leur exécutabilité et enfin calculer leur $QoiS_{moy}$.

Etape 1 : Prétraitement

Afin d'illustrer l'usage de TOPSiQC pour la configuration, nous considérons $\mathbf{S}_2$ le service de réservation de billet de train $\mathbf{S}_{\text{Réserver un billet de train}}$, présenté plus haut (*cf.* Section 9.3.1.2). Ce service ainsi que les qualités que fournit chacun de ses services atomiques sont décrits à la Figure 116. De plus, les contraintes de dépendance entre les variantes de $\mathbf{S}_{\text{Réserver un billet de train}}$ sont présentées à la Figure 118.

Nous supposons également que l'agent métier souhaite satisfaire les qualités de Commodité et de Coût, telles que présentées dans le contexte qualité de la Figure 114. Il s'agit donc de guider l'agent métier à choisir parmi les configurations de $\mathbf{S}_{\text{Réserver un billet de train}}$, celles qui se rapprochent le plus de ce qu'il souhaite satisfaire, à savoir la Commodité et le Coût avec un seuil de satisfaction égale à « + » pour chacune des qualités. L'agent métier associe la même priorité aux deux qualités (p*riority=0,5*) (*cf.* Figure 114).

GENERER LES CONFIGURATIONS PAR TOPSIQC

Comme défini plus haut (*cf.* Section 8.7.2), une matrice de décision des configurations de $\mathbf{S}_{\text{Réserver un billet de train}}$ doit être calculé. Celle-ci contient les configurations à classer ainsi que les qualités qu'elles offrent. Cette matrice est implémentée par le prototype TOPSiQC comme un vecteur dynamique. La méthode *buildTree()* du prototype TOPSiQC (*cf.* Figure 111) permet de construire (ou de générer) les différentes configurations du service agrégat de la Figure 116, tandis que la méthode *getQualityPaths()* permet de faire le calcul de la qualité de chaque configuration de ce service.

Tel que présenté à la Figure 116, le service $\mathbf{S}_{\text{Réserver un billet de train}}$ est décrit sous forme d'un arbre, sachant que les opérateurs d'agrégation « ● », « $\cup$», « v », «$\otimes$ » et « * » sont remplacés, respectivement, par *And, Xor, Or, LXor* et *Star* (*cf.* Section 8.7.2). La génération des configurations de ce service consiste donc à composer les services atomiques appartenant au service $\mathbf{S}_{\text{Réserver un billet de train}}$, et à fixer des choix par rapport aux variantes des services à variation. Comme précisé plus haut (*cf.* Tableau 17), la génération des configurations du service agrégat dépend donc de ses services composants. Par exemple, le service à choix alternatif $\otimes(\mathbf{S}_{\text{Réserver un billet de train par internet}}, \mathbf{S}_{\text{Réserver un billet de train en agence}})$ implique la génération d'une configuration qui comprendra $\mathbf{S}_{\text{Réserver un billet de train par internet}}$, et une autre qui contiendra le service $\mathbf{S}_{\text{Réserver un billet de train en agence}}$. Le service à choix multiple $v(\mathbf{S}_{\text{Présenter une carte de réduction}}, \mathbf{S}_{\text{Présenter billet de train}})$, quant à lui, implique la génération de trois configurations : une configuration qui renfermera le service $\mathbf{S}_{\text{Présenter une carte de réduction}}$, une autre comprendra le service $\mathbf{S}_{\text{Présenter billet de train}}$, et une dernière qui sera composée du service $\bullet(\mathbf{S}_{\text{Présenter une carte de}}$

réduction, $\mathbf{S}_{\text{Présenter billet de train}}$). Enfin, le service à variation de chemin $\cup(S_{\text{Annuler un billet de train}}, \bullet(\otimes(\mathbf{S}_{\text{Payer billet par carte bancaire}}, \mathbf{S}_{\text{Payer billet par chèque}}, \mathbf{S}_{\text{Payer billet par espèce}}), \nu(\mathbf{S}_{\text{Présenter une carte de réduction}}, \mathbf{S}_{\text{Présenter billet de train}}))$ génèrera deux configurations : l'une comportera $S_{\text{Annuler un billet de train}}$, alors que l'autre comprendra le service $\bullet(\otimes(\mathbf{S}_{\text{Payer billet par carte bancaire}}, \mathbf{S}_{\text{Payer billet par chèque}}, \mathbf{S}_{\text{Payer billet par espèce}}), \nu(\mathbf{S}_{\text{Présenter une carte de réduction}}, \mathbf{S}_{\text{Présenter billet de train}}))$.

Le prototype TOPSiQC génère automatiquement les différentes configurations d'un service agrégat, en l'occurrence celles du service $\mathbf{S}_{\text{Réserver un billet de train}}$ décrit à la Figure 116. En effet, à chaque élément *And*, *Xor*, *Or*, *LXor*, *Leaf* et *Star* correspond une méthode récursive, dite *agregate*, laquelle prend en entrée le vecteur de chemin *pathList* (initialisé au départ avec une chaine de caractère vide) et donne comme résultat le même vecteur modifié. La méthode *agregate* est appelée récursivement à chaque fois que l'un des éléments *And*, *Xor*, *Or*, *LXor*, *Leaf* et *Star* est rencontré. Le comportement de cette méthode dépend de ces éléments (*i.e. And*, *Xor*, *Or*, *LXor*, *Leaf* et *Star*). Par exemple, comme le montre la Figure 126, dans le cas de l'élément *And*, il s'agit pour chaque fils de ce nœud de faire appel à la méthode *agregate* (*cf.* ligne 11-14 à la Figure 126). La Figure 127 montre le cas de l'élément *LXor*, dans lequel la méthode *agregate* consiste à cloner le vecteur *pathList* en fonction du nombre de fils, et ensuite rajouter les fils dans les nouveau *pathList* (*cf.* ligne 13-21 à la Figure 127). Le *pathList* contiendra alors l'ensemble des configurations générées.

```
1   public class And extends Composite
2   {
3       ….
4   public Vector agregate (Vector pathList) throws InvalidChildException
5   {
6           //Writing the node id in the paths list
7           super.agregate(pathList);
8           //Agregate the paths from one child to another.
9           try
10          {
11              for(int i = 0; i < getChildNumber(); i++)
12              {
13                  pathList = getChild(i).agregate(pathList);
14              }
15              return pathList;
16          }catch(ClassCastException e)
17          {
18              throw new InvalidChildException();
19          }
20  }
21  }
```

Figure 126. Le calcul des configurations par TOPSiQC dans le cas de l'élément *And*

```
1   public class LXor extends Composite{
2
3       …..
4
5   public Vector agregate (Vector pathList) throws InvlidChildException
6   {
7     //Writing the node id in the paths list
8     super.agregate(pathList);
9     //Multiply the current path list with the children path.
10    try
11    {
12       Vector resPathList = new Vector();
13       for(int i = 0; i < getChildNumber(); i++)
14       {
15                  Vector multiplyPath = (Vector)pathList.clone();
16                  Vector res = getChild(i).agregate(multiplyPath);
17                  for(int j = 0; j < res.size(); j++)
18                  {
19                      resPathList.add(res.get(j));
20                  }
21       }
22  return resPathList;
23  }catch(ClassCastException e)
24          {
25              throw new InvalidChildException();
26          }
27  }
```

Figure 127. Le calcul des configurations par TOPSiQC dans le cas de l'élément *LXor*

VALIDER LES CONFIGURATIONS PAR TOPSIQC

Une fois les configurations du service $\mathbf{S}_{\text{Réserver un billet de train}}$ construites, il s'agit alors de valider leur exécutabilité, en considérant les contraintes de dépendance, définies à la Figure 118. En effet, les contraintes de dépendance entre services permettent de préciser les services qui s'excluent, ceux qui ne doivent pas faire partie de la même configuration, et ceux qui doivent impérativement se retrouver dans la même configuration. Le prototype TOPSiQC construit d'abord la liste des contraintes de dépendance *DependenceList* (*cf.* Figure 128), à partir du fichier de la Figure 118.

```
  //Building a list of the dependences
      Document docDependence = sxb.build(new File(dependencePath));
      DependenceList dependenceList  = null;
      dependenceList=
DependenceBuilder.buildDependence(docDependence.getRootElement());
```

Figure 128. Récupération des contraintes de dépendance par TOPSiQC

Ensuite, TOPSiQC vérifie la satisfaction des contraintes par chacune des configurations générées (*cf.* Figure 129). En effet, la méthode *checkDependences(path, dependenceList)* (*cf.* ligne 8 à la Figure 129) permet d'évaluer les contraintes de dépendance *dependenceList* pour chacune des configurations *path*. Cette méthode élimine les configurations non exécutables, *i.e. invalidDependence* de l'ensemble des configurations. Autrement dit, les configurations non-exécutables sont exclues.

Cette méthode recherche d'abord la (ou les) source(s) d'une contrainte dans le *path*. S'il arrive à la fin du *path* sans rencontrer de source, alors le *path* est considéré valide (*cf.* ligne 25-30 à la Figure 129), quel que soit le type de contrainte (*i.e. imply* ou *exclude*), car la contrainte ne concerne pas cette configuration, vu que la source n'étant pas présente. Si le programme rencontre la (ou les) source(s), il alors recherche dans le *path* la cible de la contrainte (*cf.* lignes 32-36 à la Figure 129). Enfin, le *path* est considéré invalide dans deux cas : (1) le programme arrive à la fin de path () et il ne trouve pas la cible, alors que le type de la contrainte est *Imply* (*i>=nodeList.length&&curDep.isImply()*, à la ligne 37 à la Figure 129) ; ou (2) le programme trouve la cible, alors que le type de la contrainte est *Exclude* (*i<nodeList.length&&!curDe.isImply()*, ligne 38-39 à la Figure 129). Nous précisons que le *path* utilisée dans cette méthode correspond à une configuration. Les sources et la cible correspondent aux services composants les contraintes de dépendance, que la configuration doit satisfaire pour qu'elle soit exécutable (*cf.* Chapitre 6).

Par exemple, le *path* peut être la configuration $\mathbf{Conf}_1 = \bullet(\mathbf{S}_{\text{Réserver un billet de train par internet}}, \mathbf{S}_{\text{Payer billet par espèce}}, \mathbf{S}_{\text{Présenter billet de train}})$ du service $\mathbf{S}_{\text{Réserver un billet de train}}$. Cette $\mathbf{Conf}_1$ n'est pas exécutable, car elle ne satisfait pas la contrainte d'exclusion suivante :$S_{\text{Réserver un billet de train par internet}} \Rightarrow \neg (S_{\text{Payer un billet de train par espèce}})$. En effet, la présence du service source $\mathbf{S}_{\text{Réserver un billet de train par internet}}$ exclut la présence du service cible $\mathbf{S}_{\text{Payer un billet de train par espèce}}$ (path invalide, car *i<nodeList.length&&!curDe.isImply()*, ligne 38-39 à la Figure 129).

```
1    //For each path in the result list.
2    for(int j = 0; j < res.size(); j++)
3     {
4       //Retrieving all the node of a path
5       String path = (String) res.get(j);
6       resTable[j][0]= path;
7       //Testing the path against the dependences
8       ArrayList<String>invalidDependences = checkDependences(path,
9   dependenceList);
10
11    …..
12
13  private ArrayList<String> checkDependences(String path,
14  DependenceList dependenceList)
15  {
16   ArrayList<String> invalidDependences = new ArrayList<String>();
17          String[] nodeList = path.split(",");
18   //Testing each dependence against the path
19  for(Object oDep : dependenceList)
20  {
21  Dependence curDep = (Dependence) oDep;
22  String[] source = curDep.getSource();
23  String target = curDep.getTarget();
24  int i = 0;
25  for(int j =0; j < source.length; j++)
26  {  while(i < nodeList.length &&
27           !cleanString(nodeList[i]).equalsIgnoreCase(source[j]))
28      {   i++;
29       }
30   }
31
32  if(i<nodeList.length)
33  {  while(i < nodeList.length &&
34       !cleanString(nodeList[i]).equalsIgnoreCase(target))
35      {   i++;
36       }
37     if((i>=nodeList.length&&curDep.isImply())||
38        (i<nodeList.length&&!curDe.isImply()))
39      {   invalidDependences.add(curDep.toString());
40       }
41  }
42  }
43  return invalidDependences;
44  }
```

Figure 129. Vérification de la satisfaction des contraintes de dépendance par TOPSiQC

CALCULER LA QOIS$_{MOY}$ DES CONFIGURATIONS EXECUTABLES PAR TOPSIQC

Une fois les configurations exécutables identifiées, la qualité moyenne QoiS$_{moy}$ des configurations est alors calculée à partir de la qualité des services atomiques, définie dans la description du service intentionnel à la Figure 116. La qualité considérée est celle indiquée dans le contexte qualité (*cf.* Figure 114) que l'agent métier souhaite satisfaire. Autrement dit, si l'agent métier souhaite satisfaire deux qualités (commodité et coût), alors que le service fournit plus (commodité, coût et performance), alors TOPSiQC considère que les qualités du contexte qualité (*i.e.* commodité et coût) et les configurations sont comparées par rapport à

ces qualités. Si une configuration ne fournit aucune des qualités que l'agent métier souhaite satisfaire, le prototype TOPSiQC affecte alors la valeur *0* à ces qualités. Ceci permet alors d'intégrer ces configurations dans le processus de comparaison, car une configuration qui est neutre (*i.e. 0*) par rapport à une qualité, peut être meilleure qu'une configuration qui offre des qualités négatives.

La $QoiS_{moy}$ est calculée dynamiquement par le prototype TOPSiQC, à partir des qualités des services atomiques, en fonction des qualités demandées par l'agent métier. La Figure 130 présente une partie de la méthode *QualityMeasure()* qui permet de faire ce calcul. Comme le montre la Figure 130, TOPSiQC calcule la somme des qualités significatives, celles qui sont différentes de « 0 » (ligne 1 à la Figure 130), des services atomiques (*leafQoSValue.intValue()*, ligne 8 à la Figure 130) composant la configuration (ligne 7-8 à la Figure 130), ainsi que le nombre des qualités différentes de « 0 » (ligne 9 à la Figure 130). Ensuite, il calcule la qualité moyenne de chaque configuration (ligne 14-15 à la Figure 130).

```
1   if(leafQoSValue.intValue() != 0)
2       {
3         if(qosValue==null
4    {qosNameList.put(cQoS.getQuality(), leafQoSValue);
5              qosCoefList.put(cQoS.getQuality(), 1);
6           }else
7   {qosNameList.put(cQoS.getQuality(),
8  qosValue.intValue()+leafQoSValue.intValue());
9  qosCoefList.put(cQoS.getQuality(), qosCoef.intValue()+1);
10          }}
11    ……….
12  //Vij Average of the agregate quality (value / coef)
13    double Vij = ((Integer)
14 qosNameList.get(qosName)).intValue()/((Integer)
15 qosCoefList.get(qosName)).intValue();
```

Figure 130. Le calcul de la qualité des configurations par TOPSiQC

Les étapes *2* à *6* de la configuration intentionnelle sont identiques aux étapes *2* à *6* de la sélection intentionnelle des services (*cf.* Section 9.3.2.1). Les mêmes programmes de TOPSiQC sont appliqués pour classer les configurations du service $\mathbf{S}_{\text{Réserver un billet de train}}$.

La Figure 131 présente le classement des configurations de $\mathbf{S}_{\text{Réserver un billet de train}}$, selon le degré de préférence C^* : celles qui d'une part possèdent la plus courte distance du contexte qualité de l'agent métier (*i.e. A**), et d'autre part ont la plus longue distance de la négation du contexte qualité (*i.e. A'*).

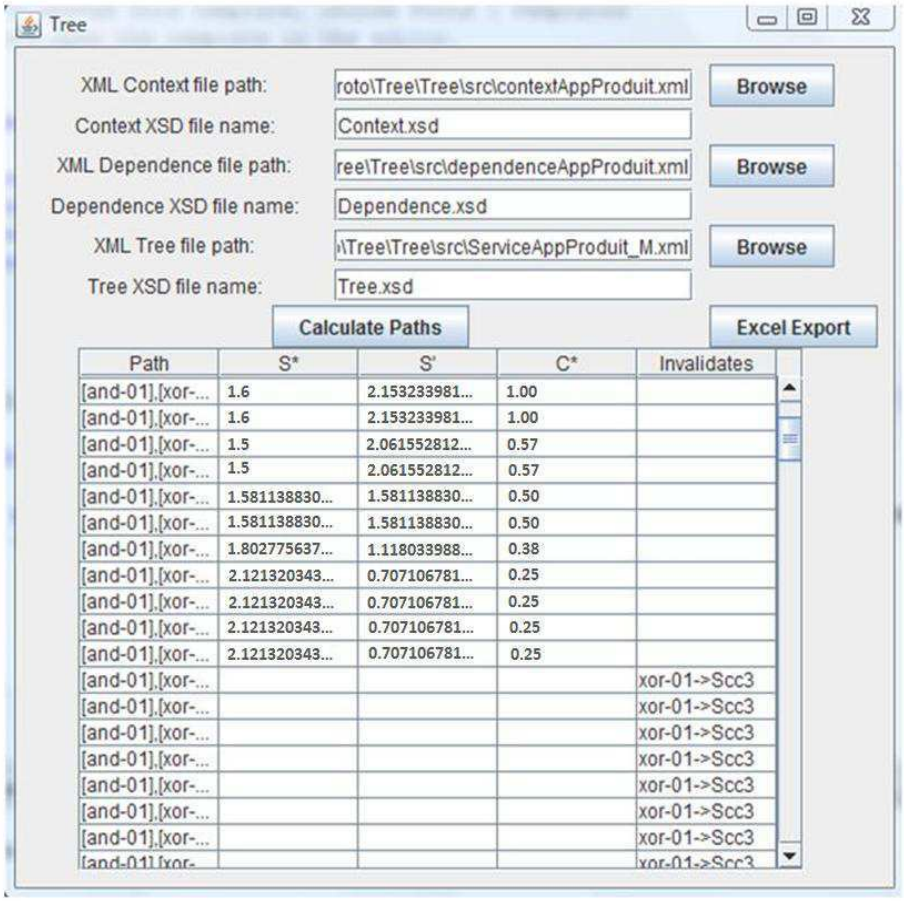

Figure 131. Les résultats de TOPSiQC dans le cas de la configuration du service $\mathbf{S}_{\text{Réserver un billet de train}}$

9.3.3 LES RESULTATS DE TOPSIQC

Les résultats fournis par le prototype TOPSiQC sont les distances S^* et S', ainsi que la proximité relative C^*. La Figure 125 met en évidence les données de sortie du prototype, dans le cas de la sélection de services. Les services $\mathbf{S}_1$, $\mathbf{S}_2$ et $\mathbf{S}_3$ sont classés, dans un ordre descendant, selon le degré de préférence C* (*i.e.* les plus proche du contexte de l'agent métier). Les résultats de TOPSiQC montrent que le service $\mathbf{S}_2$ est le plus proche du désir de l'agent métier (*cf.* Figure 125).

La Figure 131 illustre les données de sortie du prototype, dans le cas de la configuration du service $\mathbf{S}_{\text{Réserver un billet de train}}$. La Figure 131 présente les configurations du service $\mathbf{S}_{\text{Réserver un billet de train}}$ (la colonne *Path* à la Figure 131), correspondant aux différents chemins existant à l'intérieur de $\mathbf{S}_{\text{Réserver un billet de train}}$. Ces configurations sont partagées entre celles qui ne sont pas exécutables (la colonne *Invalidates* à la Figure 131), car elles ne satisfont pas les contraintes de dépendance, et les configurations exécutables (*i.e.* satisfont les contraintes de dépendance) qui sont comparées et classées, dans un ordre descendant, en fonction du degré de préférence C* (*cf.* Figure 131).

INTERPRETATION DES RESULTATS

Le prototype fournit deux types de résultat : les configurations non-exécutables, ainsi que celles qui sont exécutables. Ces dernières sont classées en fonction du contexte qualité de l'agent métier.

Les configurations non-exécutables du service $\mathbf{S}_{\text{Réserver un billet de train}}$ correspondent aux configurations qui ne satisfont pas les contraintes de dépendance. Ces configurations sont présentées afin de mettre en évidence que la variabilité du service agrégat implique, que les différentes variantes de ce service ne peuvent pas toutes être compatibles. Cette incompatibilité peut entraver l'exécution de la configuration de service, d'où la nécessité de considérer les contraintes de dépendance. Nous rappelons que les contraintes de dépendance définies dans l'exemple de la réservation de billet de train sont présentées à la Figure 118. Le prototype renvoie les configurations non-exécutables et précisent également les contraintes de dépendance non satisfaites par ces configurations (dans la colonne *Invalidates* à la Figure 131). Par exemple, la configuration **Conf**= $\bullet(\mathbf{S}_{\text{Réserver un billet de train par internet}}, \mathbf{S*}_{\text{Réserver un billet de train réduit}}, \mathbf{S}_{\text{Payer billet par espèce}}, \mathbf{S}_{\text{Présenter billet de train}})$ du service $\mathbf{S}_{\text{Réserver un billet de train}}$ n'est pas exécutable, car elle ne satisfait pas les contraintes de dépendance suivantes : $\mathbf{S}_{\text{Réserver un billet de train par internet}} \Rightarrow \neg(\mathbf{S}_{\text{Payer un billet de train par espèce}})$ et $\mathbf{S}_{\text{Réserver un billet de train réduit}} \Rightarrow (\mathbf{S}_{\text{Présenter une carte de réduction}})$.

Les configurations exécutables du service $\mathbf{S}_{\text{Réserver un billet de train}}$ correspondent aux configurations qui satisfont les contraintes de dépendance, définies à la Figure 118. Ces configurations sont comparées et sont classées en fonction du contexte qualité de l'agent métier. Dans le cas illustré à la Figure 114, ce dernier souhaite satisfaire la commodité et le coût avec le seuil de satisfaction « + » et en octroyant la même priorité aux deux qualités. Comme précisé plus haut (*cf.* Chapitre 8), les meilleures configurations sont celles qui possèdent la plus grande valeur de la proximité relative C* ; elles ont d'une part la plus petite distance du contexte qualité, et d'autre part la plus grande distance de la négation du contexte qualité.

Les résultats de TOPSiQC montrent qu'il existe des configurations du service $\mathbf{S}_{\text{Réserver un billet de train,}}$ qui correspondent complètement à ce que souhaite l'agent métier (*i.e.* C* =1). Ces meilleures configurations contiennent nécessairement le service de réservation par Internet $\mathbf{S}_{\text{Réserver un billet de train par internet}}$, car celui-ci contribue positivement (« + ») à la qualité de commodité, qualité recherchée par l'agent métier, tandis que le service Internet $\mathbf{S}_{\text{Réserver un billet de train en agence}}$ contribue négativement (« - ») à cette qualité. Aussi, ces configurations contiennent le service permettant de réserver un billet à prix réduit $\mathbf{S}_{\text{Réserver un billet de train réduit}}$, car

ce dernier contribue positivement au coût (« + »), qualité également recherchée par l'agent métier. TOPSiQC cherche donc les configurations les plus proches du souhait exprimé par l'agent métier.

9.4 *CONCLUSION*

Nous avons présenté dans ce chapitre le prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*), qui implémente l'adaptation de la méthode TOPSIS à la sélection et à la configuration intentionnelle des services (*cf.* Chapitre 8).

Le prototype TOPSiQC est développé pour montrer la faisabilité de la solution que nous avons proposée. Il permet, à partir d'une description de services, de celle des contraintes de dépendance, ainsi que celle du contexte qualité de l'agent métier, de classer, les services ou les configurations en fonction de ce contexte. L'exemple de la réservation de billet de train est utilisé pour illustrer les différentes parties de TOPSiQC.

Afin de montrer la faisabilité de notre solution, nous avons défini les données d'entrées du prototype, à savoir la description du contexte qualité de l'agent métier, celle des services, et celle des contraintes de dépendance, en utilisant XML. Ensuite, nous avons présenté l'architecture du prototype, qui se base sur le modèle MVC. Nous avons également présenté quelques morceaux de codes relatifs aux différentes étapes de la méthode (*cf.* Chapitre 8), tels que la génération des configurations, la vérification de l'exécutabilité des configurations, le calcul des distances et celui de la proximité relative. Enfin, nous avons mis en évidence les données de sortie, qui correspondent principalement au classement, dans un ordre descendant, des services ou des configurations. Le classement se fait selon la proximité relative des services par rapport au contexte qualité de l'agent métier.

Le prototype TOPSiQC peut être réutilisé dans un environnement réel pour effectuer le classement d'une multitude de services. Il peut être intégré à l'annuaire de service pour faire la sélection des services disponibles dans l'annuaire, en fonction d'un ensemble de critères de qualité. En effet, la classe *Model* qui encapsule la logique applicative de TOPSiQC peut être réutilisée dans d'autres projets où le choix entre une multitude de solutions est nécessaire.

CHAPITRE 10 : EXEMPLE DES AUTOMATES D'ANALYSE DE SANG

10.1 *INTRODUCTION*

Ce chapitre présente un exemple d'automates d'analyse de sang dans un grand laboratoire d'analyse. Ces automates appartiennent à la société Diagnostica Stago ou à d'autres sociétés du domaine du diagnostic *in vitro*. Les opérateurs du laboratoire ont besoin d'être guidés dans la sélection des automates qui répondent au mieux à leurs besoins de qualité. Particulièrement, nous nous intéressons aux automates de la société Diagnostica Stago dont le processus d'analyse est modélisé comme une famille de services qu'il faut configurer, afin de répondre au mieux aux besoins de qualité des opérateurs. Pour des raisons de confidentialité, certaines parties de cet exemple ont été adaptées. La méthodologie d'intégration de la qualité (*cf.* Chapitre 6 et Chapitre 7) et le processus de configuration intentionnelle (*cf.* Chapitre 8 et Chapitre 9) sont appliqués.

Diagnostica Stago est spécialisé dans l'analyse de l'hémostase et le diagnostic de la thrombose. Les dirigeants de la société Diagnostica Stago consacrent, aujourd'hui, tous leurs efforts de recherche et d'innovation à l'élaboration de produits et d'outils de diagnostic biologique permettant d'offrir le maximum de qualité, à savoir la précision, la sécurité, la performance, l'évolutivité…*etc*. Dans cet exemple, nous nous projetons dans le cas d'un grand laboratoire d'analyse lequel contient un nombre conséquent d'automates de la société Diagnostica Stago ou appartenant à d'autres sociétés. Dans ce cas, il s'agit de guider les opérateurs (ou les laborantins) à sélectionner parmi les automates du laboratoire, ou les configurations d'un automate donné, ceux ou celles qui répondent au mieux à leurs besoins de qualité. Ceci permettra d'optimiser l'usage des automates et répondre au mieux aux besoins des utilisateurs.

Ce chapitre est organisé comme suit. La section 2 introduit l'exemple de l'automate d'analyse de sang de la société Diagnostica Stago. La section 3 présente l'application de la méthodologie de publication de *MiS-q* au processus d'analyse de sang. La section 4 applique le prototype TOPSiQC au processus d'analyse de sang. Enfin, la section 5 conclut ce chapitre.

10.2 *EXEMPLE DE L'AUTOMATE D'ANALYSE DE SANG DE LA SOCIETE DIAGNOSTICA STAGO*

Cette section présente le contexte de l'automate d'analyse de sang de la société Diagnostica Stago et de ses principaux enjeux.

10.2.1 DIAGNOSTICA STAGO

Diagnostica Stago[18] est un groupe de 1200 personnes qui élabore, fabrique et commercialise dans le monde entier une gamme de réactifs et d'instruments d'analyses médicales permettant de dépister des maladies. La création du laboratoire Stago (Paris, France) est effectuée en 1945. En 1962, Stago met au point ses premiers réactifs d'hémostase, et en 1976, Stago se consacre exclusivement à l'hémostase et devient Diagnostica Stago. En 1979, la société crée son centre de Recherche et de Développement (R & D) des réactifs.

Avec plus de 350 produits à son catalogue, ainsi que sa présence internationale dans plus de 110 pays, Diagnostica Stago est considéré comme le leader mondial dans l'analyse de l'hémostase et le diagnostic de la thrombose. Diagnostica Stago consacre, aujourd'hui, tous ses efforts de recherche et d'innovation à l'élaboration de produits et d'outils de diagnostic biologique permettant d'offrir, à moindre coût, le maximum de qualité (précision, sécurité, performance, évolutivité, *etc.*).

10.2.1.1 *METIER DE DIAGNOSTICA STAGO*

Diagnostica Stago étudie, industrialise, fabrique commercialise et installe des dispositifs médicaux de diagnostic in vitro. Le domaine d'expertise de Stago est l'hématologie (science des maladies du sang).

La Diagnostica Stago fabrique des consommables (*e.g.* réactifs, cuvette, *etc*.) et des automates permettant de dépister des maladies relatives à la rupture de l'équilibre de la coagulation humaine, de façon automatisée. Les automates fabriqués par cette société analysent le plasma sanguin pour déterminer exactement le facteur de coagulation défectueux. Certaines substances, fabriquées par le corps humain et nécessaires au test, sont alors remplacées par des produits artificiels qui sont les réactifs.

10.2.1.2 *L'HEMOSTASE*

L'hémostase est l'ensemble des phénomènes physiologiques qui arrêtent les saignements, tout en maintenant la bonne circulation générale du sang dans les vaisseaux sanguins. Si

[18] http://www.stago.fr/fr/asp/home_global.asp

l'hémostase fonctionne mal, plusieurs maladies peuvent apparaître, telles que l'hémophilie, l'infarctus du myocarde et la phlébite.

Le sang est constitué de 55% plasma, dans lequel baignent d'autres éléments, à savoir 45% de cellules : globules rouges, globules blancs et plaquettes. La fluidité du sang (homéostasie) est maintenue grâce à un judicieux équilibre entre activateurs et inhibiteurs (antithrombine, Protéine C, Protéine S). Toute rupture de cet équilibre fera pencher la balance vers un processus pathologique : la thrombose qui est définie comme la formation de caillot pouvant provenir d'un déficit en inhibiteur, et l'hémorragie qui correspond au saignement pouvant être dû à un déficit en facteur de coagulation.

10.2.1.3 *LA FONCTION HEMOSTATIQUE*

La survenue d'une plaie vasculaire entraîne immédiatement une réaction de défense de l'organisme, visant à lutter contre le saignement (Balédent,2010). Si la plaie est peu importante, au niveau de petits vaisseaux, cette réaction peut suffire pour arrêter une hémorragie. Dans d'autres cas, la section d'une artère ou d'une veine de gros calibre, la réaction est insuffisante et seule la suture du vaisseau peut arrêter l'hémorragie. L'hémostase est composée alors de trois étapes interdépendantes : (1) l'hémostase primaire (colmatage) ; (2) la coagulation plasmatique aboutissant à la formation d'un caillot de fibrine insoluble ; et (3) la fibrinolyse permettant la dissolution du caillot de fibrine lorsque la plaie est cicatrisée. La coagulation comporte une cascade de réactions enzymatiques impliquant les facteurs de la coagulation. Les facteurs de coagulation sont désignés par des numéros allant de I à XIII.

10.2.1.4 *LES TESTS DE DIAGNOSTICA STAGO*

Deux méthodes de dosage du sang sont utilisées par les automates Stago : la méthode *chronométrique* et la méthode *photométrique* (colorimétrique et immunologique). La méthode chronométrique calcule le temps de coagulation du plasma en seconde, alors que la méthode photométrique calcule la densité optique permettant d'identifier le facteur de coagulation déficitaire.

Les valeurs normales concernant les tests de Diagnostica Stago doivent toujours être précisées afin de permettre une interprétation correcte des résultats. Diagnostica Stago possèdent plusieurs tests dont le TP (Temps Prothrombine) et le TCA (Temps de Céphaline Activée).

10.2.2 LES AUTOMATES DE DIAGNOSTICA STAGO

Les automates de Diagnostica Stago sont des instruments qui font des mesures chronométriques et photométriques. Un exemple d'automate est présenté à la Figure 132.

Figure 132. Exemple d'automate

L'automate est un instrument d'analyse, il est constitué de composants mécaniques, électroniques, et logiciels. L'automate est composé de différentes couches interdépendantes. La couche des *composants mécaniques* représente l'ensemble des équipements constituant l'automate, tels que les tiroirs, les puits de mesures, les cuvettes, les bras de pipetage et les aiguilles. La couche des *composants électroniques* représente l'ensemble des cartes électroniques permettant de faire fonctionner la partie mécanique. Elle est constituée de capteurs qui récupèrent les données concernant l'équipement, et d'actionneurs qui font bouger l'équipement. La couche *firmware* représente le logiciel embarqué qui gère les cartes électroniques, lequel est fourni par la firme qui fabrique ces cartes. Cette couche prend en considération les contraintes de temps réels qui sont obligatoire dans ce genre de système.

De plus, l'automate est composé d'un ordinateur et d'une imprimante pour éditer les résultats. L'ordinateur est doté d'un système d'exploitation permettant de gérer l'automate. Il implémente le métier de l'automate, *i.e.* le processus d'analyse de sang.

10.2.3 LA QUALITE DANS LE PROCESSUS D'ANALYSE DE SANG

Les automates de la société Diagnostica Stago, comme ceux d'autres sociétés, telles que Siemens[19] et Sysmex[20], permettent de diagnostiquer les anomalies de l'hémostase. La société

[19] http://www.siemens.com/innovation/en/publikationen/publications_pof/pof_spring_2007/molecular_medicine/vitro_diagnostics.htm

[20] http://www.sysmex.fr/index.asp?id=825

possède une large gamme d'automates (STA-compact, STA-R, *etc*.), qui réalisent des analyses de sang. Les résultats des analyses permettent de dépister l'existence d'une hémostase déficiente, laquelle prévient d'une maladie particulière, telle que l'hémophilie, la maladie de *Willebrand*, une maladie hépatique, une maladie du foie, *etc*. Ils permettent également le suivi des pathologies liées à la coagulation.

Le processus d'analyse de sang des automates est réalisé sur du plasma sanguin. Ce dernier est obtenu à partir de sang préalablement centrifugé. Nous présentons dans ce qui suit une version de ce processus. Après authentification, l'opérateur (ou le laborantin) charge, dans les tiroirs de l'automate, un ou plusieurs tubes de plasma et de produits. L'automate capte les données relatives à l'identification des tubes de plasma. L'identification du tube de plasma implique la création du dossier patient. Ensuite, l'opérateur (ou le biologiste) associe un test d'analyse (TP, TCA,…) à chacun des tubes, et lance l'analyse. Chaque test d'analyse correspond à un type de mesure : chronométrie ou photométrie. Une fois l'analyse réalisée, l'automate récupère les résultats du *firmware* et les livre à l'opérateur dans des unités brutes (secondes, densité optique/min, delta densité optique). Les résultats sont également exprimés dans des unités plus compréhensibles (par exemple %) pour les opérateurs. En cas de problèmes ou de résultats erronés (par exemple résultat hors limites autorisées), l'automate relance les analyses.

Par ailleurs, le GEHT (Groupe d'Etudes sur l'Hémostase et la Thrombose)[21] a identifié, un ensemble de qualités que tous les fabricants, de ce type d'instruments d'analyse, doivent respecter, telles que la performance en termes de cadence (nombre de tubes réalisés par jour), l'interopérabilité (communication des automates avec les systèmes d'information des hôpitaux et/ou des laboratoires d'analyse), *etc*. Ainsi, les automates doivent plus ou moins fournir ces qualités.

Dans cet exemple, nous nous projetons dans le cas d'un grand laboratoire d'analyse, qui contient un nombre conséquent d'automates de la société Diagnostica Stago ou appartenant à d'autres sociétés. Il s'agit de guider les opérateurs (ou les laborantins) à sélectionner les automates du laboratoire qui répondent au mieux à leurs besoins de qualité. Particulièrement, nous proposons de configurer le processus d'analyse de sang des automates de la société Diagnostica Stago, afin de sélectionner les configurations qui répondent au mieux aux besoins de qualité des opérateurs. Par exemple, selon la demande d'analyse du moment, les opérateurs

[21] http://site.geht.org/site/Le-GEHT_14_.html

auront besoin plus d'un automate performant que d'un automate sécurisé, *etc*. Ceci permettra de répondre au mieux aux besoins des utilisateurs (*i.e.* les opérateurs ou les laborantins).

Pour cela, nous proposons d'utiliser la méthodologie de sélection et de configuration intentionnelle des services, proposée dans cette thèse. Notre solution est centrée utilisateur, avec des descriptions orientées but des services et de leurs qualités. Afin, de guider les opérateurs dans la sélection et la configuration des automates, qui répondent au mieux à leurs exigences non-fonctionnelles, nous proposons d'utiliser le prototype TOPSiQC (*cf.* Chapitre 9), lequel supporte la sélection et la configuration intentionnelle des services (*cf.* Chapitre 8). Avant d'appliquer le prototype TOPSiQC à notre exemple, nous proposons de modéliser d'abord le processus d'analyse du plasma sanguin, brièvement décrit dans la section précédente, en appliquant la méthodologie de publication du modèle *MiS-q* (*cf.* Chapitre 7).

10.3 *APPLICATION DE LA METHODOLOGIE DE PUBLICATION DE MiS-q*

Nous proposons d'appliquer la méthodologie de publication du modèle *MiS-q*, laquelle est détaillée au Chapitre 7. Celle-ci consiste à construire d'abord la carte des besoins correspondant au processus d'analyse de sang, et d'intégrer la qualité à cette carte. Ensuite, générer les services intentionnels et leurs QoiS, opérationnaliser le service atomique, et enfin associer des contraintes de dépendance au service agrégat. Telle que présentée au Chapitre 7, cette méthodologie se décline en plusieurs étapes :

1. Etape1 : intégration de la qualité à la carte des besoins
2. Etape2 : génération de la QoiS simple
3. Etape 3 : opérationnalisation des services atomiques
4. Etape 4 : génération de la QoiS globale
5. Etape 5 : association des contraintes de dépendance au service intentionnel agrégat

Nous proposons d'appliquer chacune de ces étapes au processus d'analyse de sang, à savoir la carte *Réaliser analyse plasma*.

10.3.1 ETAPE1 : INTEGRATION DE LA QUALITE A LA CARTE REALISER ANALYSE PLASMA

La première étape consiste à intégrer la qualité à la carte *Réaliser analyse plasma*. Comme présenté au Chapitre 7, l'intégration de la qualité se fait comme suit :

1. Construire la carte *Réaliser analyse plasma*
2. Consulter le référentiel qualité du diagnostic *in vitro*
3. Identifier les buts qualité de la carte *Réaliser analyse plasma*

4. Intégrer les buts qualité à la carte *Réaliser analyse plasma*

10.3.1.1 *CONSTRUIRE LA CARTE* REALISER ANALYSE PLASMA

La Figure 133 présente la carte correspondant au processus d'analyse de sang. Tel que présenté au Chapitre 7, la carte de la Figure 133 est décrite en termes intentionnels. Elle permet d'atteindre le but *Réaliser analyse plasma* et elle correspond au service agrégat $\mathbf{S}_{\textit{Réaliser analyse plasma}}$. Comme le montre la Figure 133, les buts principaux assignés au processus d'analyse du plasma sanguin sont *Démarrer*, *Charger les tubes*, *Identifier les tubes*, *Récupérer plasma*, *Traiter plasma*, *Analyser plasma*, *Mesurer plasma* et *Arrêter*, qui sont codés respectivement par *a*, *b*, *c*, *d*, *e*, *f*, *g* et *h* (voir la codification de la carte dans le Chapitre 7). Un ensemble de stratégies sont définies afin de préciser les différentes manières d'accomplir ces buts. Les buts, les stratégies ainsi que les services associés à la carte *Réaliser analyse plasma* sont décrits dans ce qui suit. Cette carte est construite en se basant sur notre connaissance du domaine, ainsi que sur le travail présenté par (Djebbi,2011).

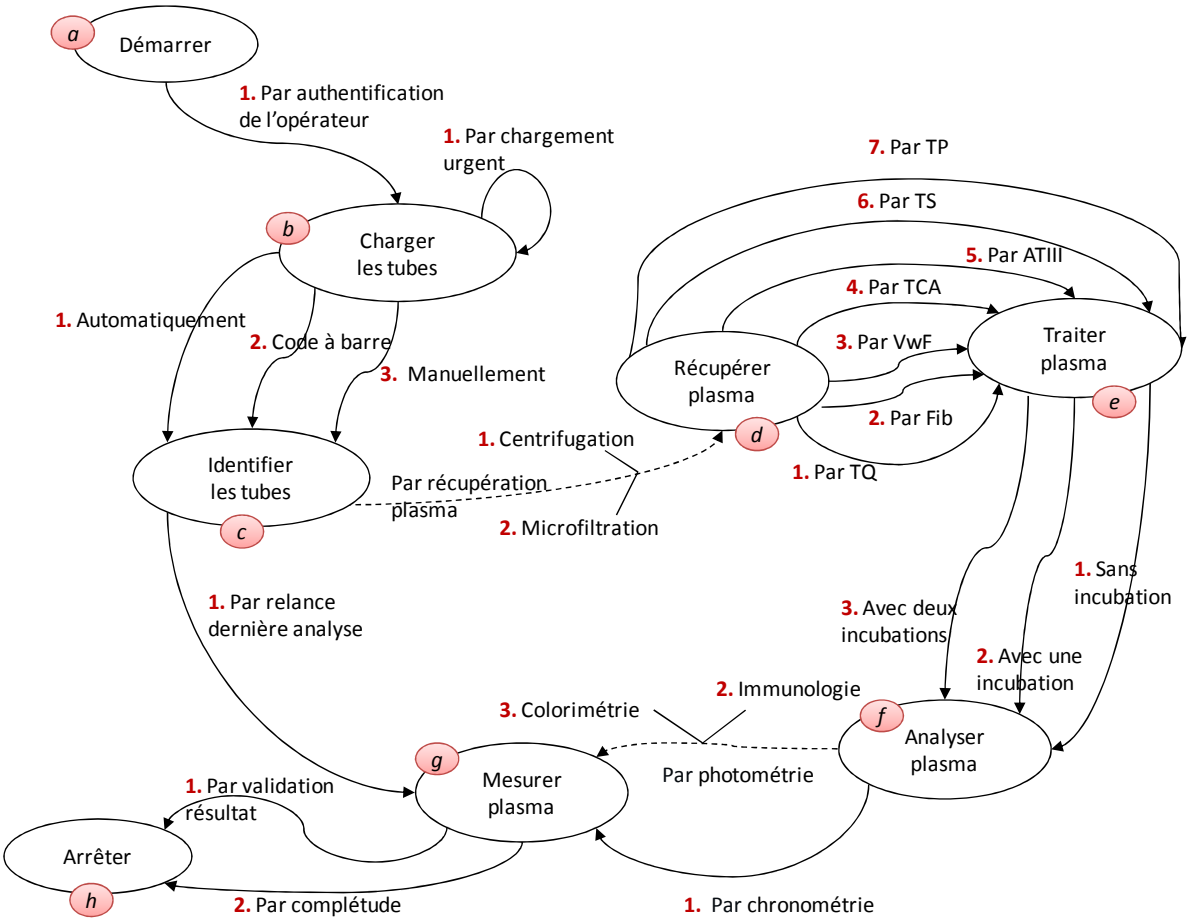

Figure 133. La carte *Réaliser analyse plasma*

Les buts de la carte *Réaliser analyse plasma* sont *Démarrer*, *Charger les tubes*, *Identifier les tubes*, *Récupérer plasma*, *Traiter plasma*, *Analyser plasma*, *Mesurer plasma* et *Arrêter*.

Ces buts correspondent aux étapes que l'opérateur peut souhaiter (ou doit obligatoirement) effectuer pour réaliser une analyse de plasma sanguin.

- *« Démarrer »* indique le point de départ du processus.
- *« Charger les tubes »* fait référence au chargement des tubes échantillons à analyser. Le chargement des tubes consiste d'abord à authentifier l'opérateur grâce à la stratégie *Par authentification de l'opérateur* (codé *ab1* à la Figure 133). L'authentification permet de sécuriser l'accès aux données personnelles des patients. Une fois l'opérateur authentifié, le chargement par défaut des tubes est effectué en mode batch. Néanmoins, un automate peut prendre en charge des tubes urgents (codée *bb1* à la Figure 133).
- *« Identifier les tubes »* exprime que chaque tube de sang chargé dans l'automate est identifié par un code unique. Le code peut être attribué automatiquement (un numéro incrémental) (*i.e.* la stratégie *Automatiquement*, codée *bc1* à la Figure 133), introduit par l'opérateur (*i.e.* la stratégie *Manuellement*, codée *bc3* à la Figure 133), ou lu à travers un code à barre (*i.e.* la stratégie *Code à barre*, codée *bc2* à la Figure 133) se trouvant sur le tube.
- *« Récupérer plasma »* correspond d'abord à séparer le plasma des différents constituants du sang (plaquettes et globules rouges et blanc). En effet, une fois les tubes de sang, servant à l'analyse, chargés par l'opérateur dans les tiroirs de l'instrument, deux techniques peuvent être utilisées pour récupérer le plasma : la centrifugation[22] (codée *cd1* à la Figure 133) ou la microfiltration (codée *cd2* à la Figure 133).
- *« Traiter plasma »* correspond à assigner un ou plusieurs tests d'analyse aux tubes. Les tests de laboratoire supportés par les automates Stago sont : Temps de saignement (TS) (codée *de6* à la Figure 133), Dosage du facteur de Von Willebrand (VwF) (codée *de3* à la Figure 133), Temps de céphaline activé (TCA) (codée *de4* à la Figure 133), Taux de prothrombine (TP) (codée *de7* à la Figure 133), Temps de Quick (TQ) (codée *de1* à la Figure 133), Dosage des facteurs de coagulation : ATIII (codée *de5* à la Figure 133) et Fib (codée *de2* à la Figure 133).
- *« Analyser plasma »* correspond à appliquer une méthodologie de test, laquelle décrit un ordonnancement de tâches, telles que le pipetage, la dilution, l'ajout de réactif ou l'incubation. L'automate peut implémenter une méthodologie à deux incubations (codée *ef3* à la Figure 133), à une seule incubation (codée *ef2* à la Figure 133) et/ou sans incubation (codée *ef1* à la Figure 133).

[22] Soumission à un mouvement de rotation rapide à l'ordre de 2500 tours/seconde.

- *« Mesurer plasma »* correspond à appliquer à chaque test biologique une des méthodes de mesure suivantes : la mesure chronométrique (*i.e.* la stratégie *Par chronométrie*, codée *fg1* à la Figure 133) ou la mesure photométrique, qui peut être immunologique (codée *fg2* à la Figure 133) ou colorimétrique (codée *fg3* à la Figure 133). La méthode chronométrique sert à mesurer le temps que prend le sang pour coaguler. Techniquement, sur l'automate, la cuvette dans laquelle le plasma est prélevé, contient une petite bille en acier. Après traitement (dilution, incubation, *etc*.), la bille est agitée au moyen d'une bobine magnétique. Le moment où le sang se coagule correspond au moment où la bille s'arrête de s'agiter. La mesure rendue est le temps de coagulation. La mesure photométrique, quant à elle, consiste à mesurer la densité optique du sang contenu dans la cuvette, appelée DDO (Delta de Densité Optique). La différence entre la mesure immunologique et la mesure colorimétrique réside dans la longueur d'onde utilisée (540 nm et 405 nm respectivement). Une fois l'analyse réalisée, si les résultats obtenus sont erronés, il est alors possible de relancer la dernière analyse ultérieurement grâce à la stratégie *Par relance dernière analyse* (codée *cg1* à la Figure 133).
- *« Arrêter »* indique le point d'arrêt du processus d'analyse. Le processus d'analyse se termine soit par complétude soit par validation des résultats. Dans le premier cas, les résultats obtenus sont acceptés sans aucune validation de l'opérateur. L'automate arrête ainsi le processus via la stratégie *Complétude* (codée *gh2* à la Figure 133). Dans le dernier cas, le processus se termine par une validation des résultats par l'opérateur, via la stratégie *Par validation résultat* (codée *gh1* à la Figure 133).

10.3.1.2 *CONSULTER LE REFERENTIEL QUALITE DU DIAGNOSTIC IN VITRO*

Nous avons proposé de définir, par domaine d'activité, un référentiel qualité (*cf.* Chapitre 5). Le référentiel qualité permet de fournir consensus sur la qualité, dans un domaine donné.

Dans le domaine de l'hémostase, le GEHT (Groupe d'Etudes sur l'Hémostase et la Thrombose)[23] est un groupe d'étude de la Société Française d'Hématologie. Le GEHT réunit les universitaires, les praticiens hospitaliers, les chercheurs des secteurs publics ou privés, ainsi que les membres de l'industrie pharmaceutique ou du diagnostic (SFRL- Société de l'industrie du diagnostic *in vitro*), qui œuvrent dans le domaine de l'Hémostase et de la Thrombose.

Dans ce domaine, le GEHT a pour mission d'aider la recherche fondamentale et clinique, d'améliorer la pratique médicale et biologique et de promouvoir l'innovation industrielle. Le

[23] http://site.geht.org/site/Le-GEHT_14_.html

GEHT s'intéresse au diagnostic *in vitro* de l'hémostase en fournissant une standardisation des méthodologies d'analyse, ainsi que le contrôle de qualité. Par ailleurs, le GEHT (GEHT,2010) a identifié, suite à l'élaboration d'un questionnaire, les critères de choix d'un automate, tels que la performance de l'automate, l'interopérabilité de l'automate, la confidentialité des données et la facilité d'utilisation de l'automate. Ces critères doivent être respectés par tous les fabricants de ce type d'instruments d'analyse.

La conception d'un référentiel qualité dans le domaine du diagnostic *in vitro* de l'hémostase s'avère donc nécessaire, afin de permettre aux différents fabricants d'automate, de partager les critères de choix définis par le GEHT, ainsi que les méthodologies permettant de les satisfaire. Il permet également aux utilisateurs de l'automate de prendre connaissance des qualités que peuvent fournir les automates. Par exemple, la Figure 134 illustre le référentiel qualité que nous proposons pour le but qualité Performance. La performance dans le domaine du diagnostic *in vitro* de l'hémostase est définie en termes de Temps. Celle-ci est quantifiée par la cadence, qui est exprimée en nombre de tubes effectués par jour. Par exemple, les automates sont considérés très performants lorsque leur cadence dépasse *120* tubes/jour. De plus, un ensemble de buts, permettant de satisfaire partiellement la cadence, dans le domaine du diagnostic *in vitro* de l'hémostase, est recommandé. Par exemple, la réalisation de l'analyse par chronométrie contribue à la satisfaction « + » du temps, tandis que la réalisation de l'analyse par photométrie contribue à la satisfaction « - ».

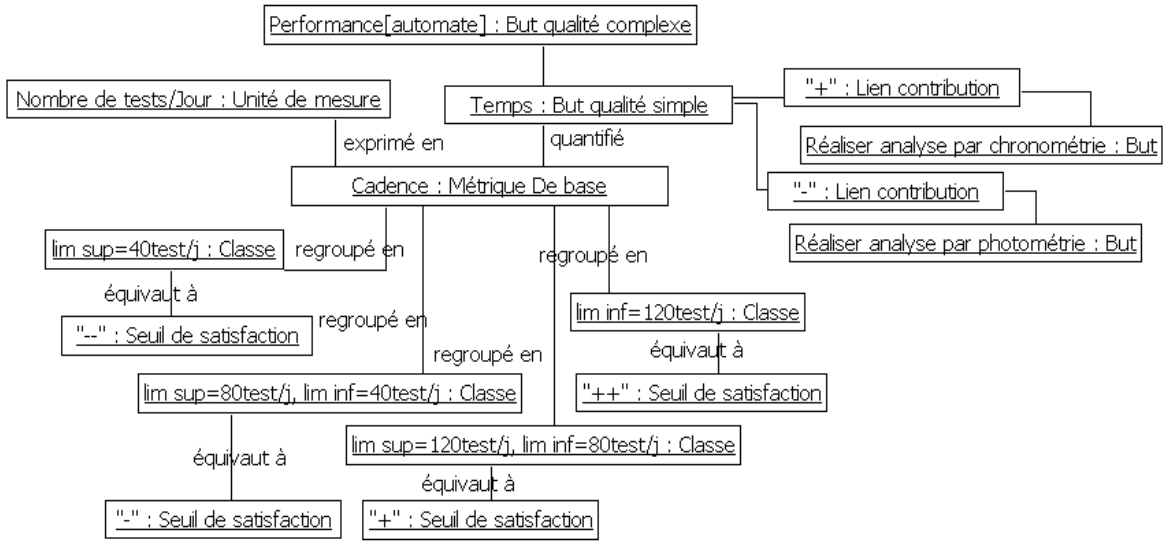

Figure 134. Le but qualité Performance dans le domaine du diagnostic *in vitro* de l'hémostase

Les fabricants d'automate, tel que la société Diagnostica Stago, consulte alors le référentiel qualité relatif au domaine du diagnostic *in vitro* de l'hémostase. Ensuite, ils choisissent les

buts qualité auxquels les services de leurs automates (*i.e.* la carte *Réaliser analyse plasma*) contribuent à satisfaire partiellement, ainsi que les seuils de satisfaction correspondants. Ces derniers sont définis « *à priori* » en prenant comme référence ceux du référentiel qualité.

10.3.1.3 *IDENTIFIER LES BUTS QUALITE DE LA CARTE REALISER ANALYSE PLASMA*

Afin d'illustrer notre démarche, nous supposons que la société Diagnostica Stago considère trois qualités définies par le GEHT, à savoir la sécurité en termes de confidentialité, la précision et la performance en termes de temps. En effet, l'automate doit assurer la confidentialité des données personnelles du patient. Il doit également assurer la précision des données saisies et des résultats, et enfin il doit assurer la performance. Ces qualités sont alors intégrées à la carte Réaliser analyse plasma.

10.3.1.4 *INTEGRER LES BUTS QUALITE A LA CARTE REALISER ANALYSE PLASMA*

Les qualités identifiées préalablement (*i.e.* la confidentialité, la précision et le temps) sont alors associées aux sections de la carte *Réaliser analyse plasma* (*cf.* Section 10.3.1.3). La Figure 135 met en évidence les liens de contribution reliant la carte *Réaliser analyse plasma* aux buts qualité Confidentialité, Précision et Temps.

Comme nous l'avons présenté au Chapitre 7, la carte permet de satisfaire dans des limites acceptables les buts qualité. Les qualités et les seuils de satisfaction, présentés dans cette section, sont définis dans le référentiel qualité du domaine du diagnostic *in vitro* de l'hémostase (*cf.* Section 10.3.1.3). Toutes les sociétés qui fabriquent ce type d'automates pourront consulter ce référentiel qualité, afin d'ajuster la contribution de leurs services aux diverses qualités existantes. Ainsi, les buts qualité de Confidentialité, de Précision et de Temps, identifiés à partir du référentiel, peuvent être satisfait (« + »), très satisfait (« ++ »), peu satisfait (« - ») ou pas du tout satisfait (« -- ») par les sections de la carte.

- Confidentialité : l'automate doit assurer la confidentialité des données personnelles du patient. Pour cela, il fournit un service d'authentification permettant ainsi de contrôler l'accès aux opérateurs souhaitant manipuler les données du patient (*e.g.* tube plasma, résultat). Ceci est effectué grâce à la stratégie *Par authentification de l'opérateur*. La section *Authentifier l'opérateur* (codée *ab1* à la Figure 135) contribue alors à une satisfaction « + » de la Confidentialité.
- Précision : l'automate doit assurer la précision des données saisies, ainsi que les résultats obtenus. En effet, il doit assurer la précision des données saisies, en proposant une identification automatique (*i.e.* la section *Identifier tubes automatiquement* à la Figure 135) ou par code à barre (*i.e.* la section *Identifier tubes par code à barre* à la Figure 135) des

tubes, qui contribue à une satisfaction « + » de la Précision. Néanmoins, l'identification manuelle (*i.e.* la section *Identifier tubes manuellement* à la Figure 135) contribue à la satisfaction « - » de la Précision. Egalement, la microfiltration (*i.e.* la section *Récupérer plasma par microfiltration* à la Figure 135) est un procédé permettant de fournir des résultats plus précis que la centrifugation (*i.e.* la section *Récupérer plasma par centrifugation* à la Figure 135). La relance d'une analyse de plasma sanguin diminue fortement la précision des résultats. Donc, la stratégie *Par relance dernière analyse* (codée *cg1* à la Figure 135) contribue à une satisfaction « -- » de la Précision. La validation des résultats par l'opérateur renforce la précision des résultats. En effet, si les résultats obtenus sont validés grâce à la stratégie *Par validation résulta* (codée *gh1* à la Figure 135), alors celle-ci contribue à une satisfaction « + » de la Précision. Enfin, la section *Arrêter* p*ar complétude* (codée *gh2* à la Figure 135) contribue à une satisfaction « - » de la Précision, car les résultats ne sont pas validés par l'opérateur.

- Temps : l'automate doit assurer la performance, en termes de temps, qui est mesurée par la cadence. L'authentification de l'opérateur (codée *ab1* à la Figure 135) contribue « - » à la satisfaction du Temps. L'identification automatique (codée *bc1* à la Figure 135) contribue à une satisfaction « + » du Temps, alors que l'identification par code à barre (codée *bc2* à la Figure 135) et l'identification manuelle (codée *bc3* à la Figure 135) des tubes contribuent à une satisfaction « - » du Temps. Egalement, la microfiltration (codée *cd2* à la Figure 135) est un procédé plus lent (*i.e.* contribue « - » au Temps) que la centrifugation (codée *cd1* à la Figure 135) (*i.e.* contribue « + » au Temps). Le choix de la méthodologie d'analyse influence également le temps. En effet, l'analyse sans incubation (codée *ef1* à la Figure 135), avec une (codée *ef2* à la Figure 135) ou deux incubations (codée *ef3* à la Figure 135) contribuent, respectivement, à « + », « - » et « -- » au Temps. Le choix de la méthode de mesure (chronométrie ou photométrie) impacte le Temps : l'utilisation de la *méthode de mesure chronométrique, colorimétrique* et *immunologique* contribuent respectivement à une satisfaction « + », « - » et « - » du Temps. La relance de l'analyse (codée *cg1* à la Figure 135) contribue à une satisfaction « -- » du Temps, car le temps supplémentaire de la relance diminue nécessairement le nombre de tests à effectuer par jour. Enfin, la validation des résultats (codée *gh1* à la Figure 135) contribue à une satisfaction « - » du Temps, alors que la section *Arrêter* p*ar complétude* (codée *gh2* à la Figure 135) contribue à une satisfaction « + » du Temps.

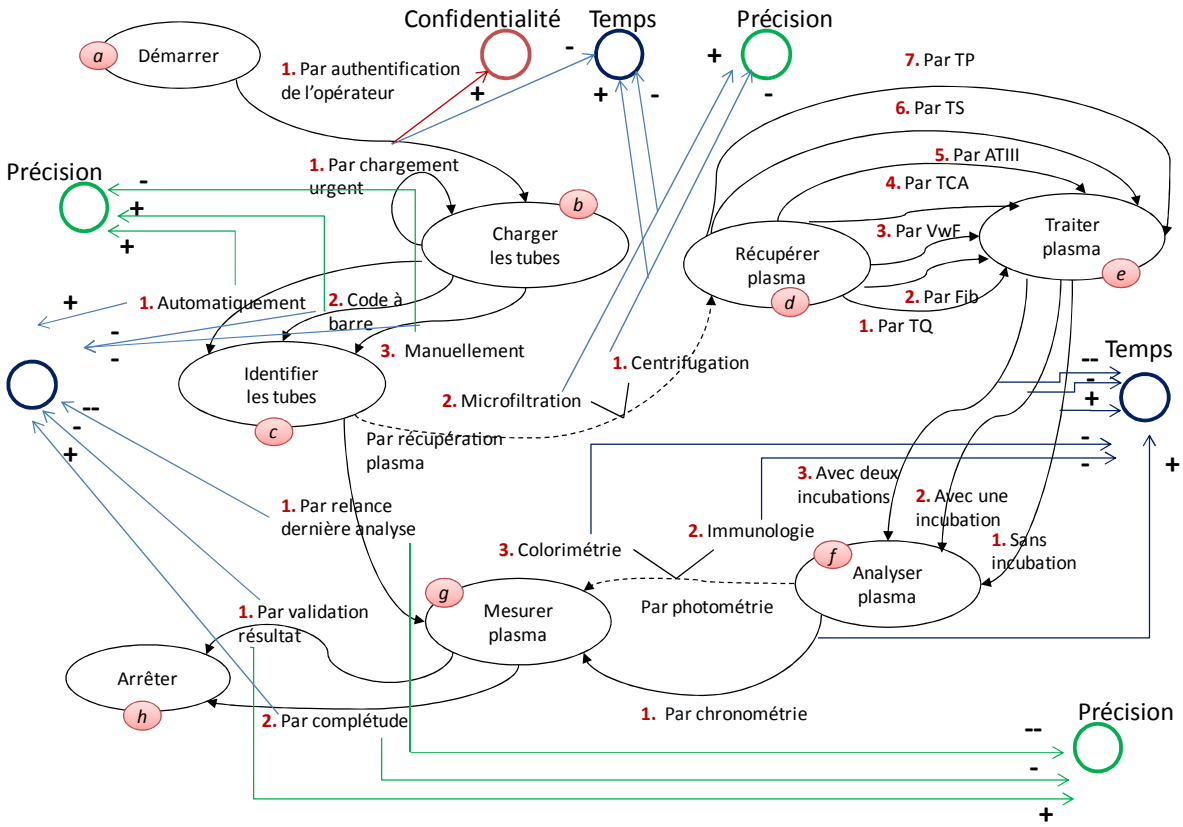

Figure 135. Intégration de la qualité à la carte *Réaliser analyse plasma*

Une fois la carte des besoins construite et enrichie avec les qualités, les services intentionnels peuvent alors être générés, ainsi que leurs qualités.

10.3.2 ETAPE 2 : GENERATION DE LA QOIS SIMPLE DE LA CARTE REALISER ANALYSE PLASMA

La deuxième étape consiste à générer les différents services intentionnels de la carte *Réaliser analyse plasma*, ainsi que la QoiS simple. Cette étape consiste à :

1. Générer les services intentionnels de la carte *Réaliser analyse plasma*
2. Générer la QoiS simple des services atomiques de la carte *Réaliser analyse plasma*

10.3.2.1 *GENERER LES SERVICES INTENTIONNELS DE LA CARTE REALISER ANALYSE PLASMA*

Tel que défini plus haut, dans l'architecture iSOA, pour établir un couplage direct entre les buts métier et les fonctionnalités du système, les sections de la carte et les relations entre ces sections sont traduites en des services intentionnels (*cf.* Chapitre 4). A partir de buts et des stratégies identifiés ci-dessus (*cf.* Figure 133), nous pouvons générer un certain nombre de services intentionnels associés à la carte *Réaliser analyse plasma*. Nous présentons ci-dessous les services intentionnels atomiques et agrégats.

10.3.2.1.1 GENERER LES SERVICES ATOMIQUES

Nous associons, comme défini au Chapitre 4, à chaque section opérationnalisable de la carte un service atomique. La carte de la Figure 133 est alors composée de 24 services atomiques. Le Tableau 26 résume les services atomiques associés à la carte *Réaliser analyse plasma*.

Tableau 26. Les services atomiques associés à la carte *Réaliser analyse plasma*

Section de la carte	Service atomique
ab1	$S_{\text{Authentifier l'opérateur}}$
bb1	$S^*_{\text{Charger les tubes par mode urgent}}$
bc1	$S_{\text{Identifier tubes automatiquement}}$
bc2	$S_{\text{Identifier tubes par code à barre}}$
bc3	$S_{\text{Identifier tubes manuellement}}$
cg1	$S_{\text{Relancer dernière analyse}}$
cd1	$S_{\text{Récupérer plasma par centrifugation}}$
cd2	$S_{\text{Récupérer plasma par microfiltration}}$
de1	$S_{\text{Traiter plasma par }\mathbf{TQ}}$
de2	$S_{\text{Traiter plasma par }\mathbf{Fib}}$
de3	$S_{\text{Traiter plasma par }\mathbf{VwF}}$
de4	$S_{\text{Traiter plasma par }\mathbf{TCA}}$
de5	$S_{\text{Traiter plasma par }\mathbf{ATIII}}$
dd6	$S_{\text{Traiter plasma par }\mathbf{TS}}$
de7	$S_{\text{Traiter plasma par }\mathbf{TP}}$
ef1	$S_{\text{Analyser plasma sans incubation}}$
ef2	$S_{\text{Analyser plasma avec une incubation}}$
ef3	$S_{\text{Analyser plasma avec deux incubations}}$
fg1	$S_{\text{Réaliser l'analyse par mesure chronométrique}}$
fg2	$S_{\text{Réaliser l'analyse par mesure immunologique}}$
fg3	$S_{\text{Réaliser l'analyse par mesure colorimétrique}}$
gh1	$S_{\text{Arrêter par validation résultat}}$
gh2	$S_{\text{Arrêter par complétude}}$

10.3.2.1.2 GENERER LES SERVICES AGREGATS

Nous associons, comme présenté dans le Chapitre 4, à chaque relation, entre les sections de la carte, un service agrégat. Les services agrégats sont définis en se basant sur l'algorithme de MacNaughton et Yamada (MacNaugthon&Yamada,1960) (*cf.* Chapitre 4). En effet, ce dernier est appliqué à la carte de la Figure 133, afin de définir tous les chemins possibles entre le but *Démarrer* et le but *Arrêter*. La formule, décrivant tous les chemins reliant le but *a* au but *d*, est définie comme suit (*cf.* Chapitre 4) :

$$Y_{a,\{a,b,c,d\},d} = \bullet(X_{ab}, X^*_{bb}, \cup (X_{bd}, \bullet (X_{bc}, X^*_{cc}, X_{cd}))) \qquad (1)$$

Cette formule indique que pour aller de *a* à *d*, il y a une séquence à suivre. En effet, la formule X_{ab} est utilisée pour satisfaire le but *b* à partir du but *a*. Ensuite, X^*_{bb} est utilisée pour aller de *b* vers *b*. Pour aller de *b* à *d*, il faut choisir l'un des deux chemins suivants : soit aller

de *b* à *d* directement (par la formule X_{bd}), soit aller de *b* à *c* puis de *c* à *c* et enfin de *c* à *d* (par la formule ● (X_{bc}, X^*_{cc}, X_{cd})).

En appliquant l'algorithme décrit dans le Chapitre 4 à la formule *(1)*, nous obtenons le service agrégat qui réalise le but *Réaliser analyse plasma*, à savoir $\mathbf{S}_{\text{Réaliser analyse plasma}}$. Ainsi, la formule *(1)* devient *(2)* :

$$\mathbf{S}_{\text{Réaliser analyse plasma}} = \bullet\ (\mathbf{S}_{ab1},\ \mathbf{S}^*_{bb1},\ \nu\ (\mathbf{S}_{bc1},\mathbf{S}_{bc2},\ \mathbf{S}_{bc3}),\ \cup(\mathbf{S}_{cg1},\ \bullet(\otimes(\mathbf{S}_{cd1},\ \mathbf{S}_{cd2}), \qquad (2)$$

$$\nu\ (\mathbf{S}_{de1},\mathbf{S}_{de2},\ \mathbf{S}_{de3},\mathbf{S}_{de4},\mathbf{S}_{de5},\mathbf{S}_{de6},\mathbf{S}_{de7}),\ \nu\ (\mathbf{S}_{ef1},\mathbf{S}_{ef2},\ \mathbf{S}_{ef3}),\ \nu\ (\mathbf{S}_{fg1},\ \otimes(\mathbf{S}_{fg2},\ \mathbf{S}_{fg3})))),$$

$$\nu\ (\mathbf{S}_{gh1},\ \mathbf{S}_{gh2}))$$

Comme défini dans le Chapitre 4, à la carte des besoins peut correspondre plusieurs types de services, à savoir le service à choix alternatif (noté par « $\otimes$ »), le service à choix multiple (noté par « ν »), le service séquentiel (noté par « ● »), et le service à variation de chemin (noté par « $\cup$ »). Chacun de ces services dépend du type de relation entre les sections de la carte, à savoir le paquet, le multi-segment, le chemin et le multi-chemin. Le Tableau 27 résume les différents types de services agrégats associés à la carte *Réaliser analyse plasma*. Le Tableau 27 précise également les relations dans la carte correspondant à ces services. Ainsi, les services fournis par l'automate sont décrits dans des termes compréhensibles par les utilisateurs (décideurs, biologiste, opérateur, clients) de l'automate, car les services intentionnels correspondent aux exigences fonctionnelles de ces utilisateurs. La Figure 136 illustre la représentation graphique du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$.

Les sections de la carte de la Figure 133 sont toutes opérationnalisables. Néanmoins, dans le cas de section non-opérationnalisable (à un niveau d'abstraction *i*), celle-ci est affinée par une carte complète (de niveau d'abstraction *i+1*) (*cf.* Chapitre 4). A cette carte est appliqué l'algorithme de MacNaughton et Yamada afin de définir la formule du service agrégat lui correspondant. Le résultat obtenu au niveau *i+1* est alors déplacé au niveau *i*.

Tableau 27. Les services agrégats associés à la carte *Réaliser analyse plasma*

Service agrégat		**Relation dans la carte**
Service à choix alternatif	$\mathbf{S}_{\text{Récupérer plasma}} = \otimes(\mathbf{S}_{\text{Récupérer plasma par centrifugation}}, \mathbf{S}_{\text{Récupérer plasma par microfiltration}})$	$\otimes(\mathbf{S}_{cd1}, \mathbf{S}_{cd2})$
	$\mathbf{S}_{\text{Mesurer plasma par photométrie}} = \otimes(\mathbf{S}_{\text{Mesurer plasma par immunologie}}, \mathbf{S}_{\text{Mesurer plasma par colorimétrie}})$	$\otimes(\mathbf{S}_{fg2}, \mathbf{S}_{fg3})$
Service à choix alternatif	$\mathbf{S}_{\text{Identifier tubes}} = \nu(\mathbf{S}_{\text{Identifier tubes automatiquement}}, \mathbf{S}_{\text{Identifier tubes par code à barre}}, \mathbf{S}_{\text{Identifier tubes manuellement}})$	$\nu(\mathbf{S}_{bc1}, \mathbf{S}_{bc2}, \mathbf{S}_{bc3})$
	$S_{\text{Traiter plasma par choix test}} = \nu(S_{\text{Traiter plasma par TQ}}, S_{\text{Traiter}}$	$\nu(\mathbf{S}_{de1}, \mathbf{S}_{de2}, \mathbf{S}_{de3}, \mathbf{S}_{de4}, \mathbf{S}_{de5}, \mathbf{S}_{de6}, \mathbf{S}_{de7})$

	plasma par Fib, $S_{\text{Traiter plasma par VwF}}$, $S_{\text{Traiter plasma par TCA}}$, $S_{\text{Traiter plasma par ATIII}}$, $S_{\text{Traiter plasma par TS}}$, $S_{\text{Traiter plasma par TP}}$)	
	$S_{\text{Analyser plasma}} = \vee(S_{\text{Analyser plasma sans incubation}}, S_{\text{Analyser plasma avec une incubation}}, S_{\text{Analyser plasma avec deux incubations}})$	$\vee (S_{ef1}, S_{ef2}, S_{ef3})$
	$S_{\text{Mesurer plasma}} = \vee(S_{\text{Mesurer plasma par chronométrie}}, S_{\text{Mesurer plasma par photométrie}})$	$\vee (S_{fg1}, \otimes (S_{fg2}, S_{fg3}))$
	$S_{\text{Arrêter analyse}} = \vee(S_{\text{Arrêter par validation résultat}}, S_{\text{Arrêter par complétude}})$	$\vee (S_{gh1}, S_{gh2})$
Service séquentiel	$S_{\text{Réaliser analyse plasma}}$	$\bullet(S_{ab1}, S^*_{bb1}, \vee (S_{bc1}, S_{bc2}, S_{bc3}), \cup(S_{cg1}, \bullet(\otimes (S_{cd1}, S_{cd2}), \vee (S_{de1}, S_{de2}, S_{de3}, S_{de4}, S_{de5}, S_{de6}, S_{de7}), \vee (S_{ef1}, S_{ef2}, S_{ef3}), \vee (S_{fg1}, \otimes (S_{fg2}, S_{fg3}))), \vee (S_{gh1}, S_{gh2}))$

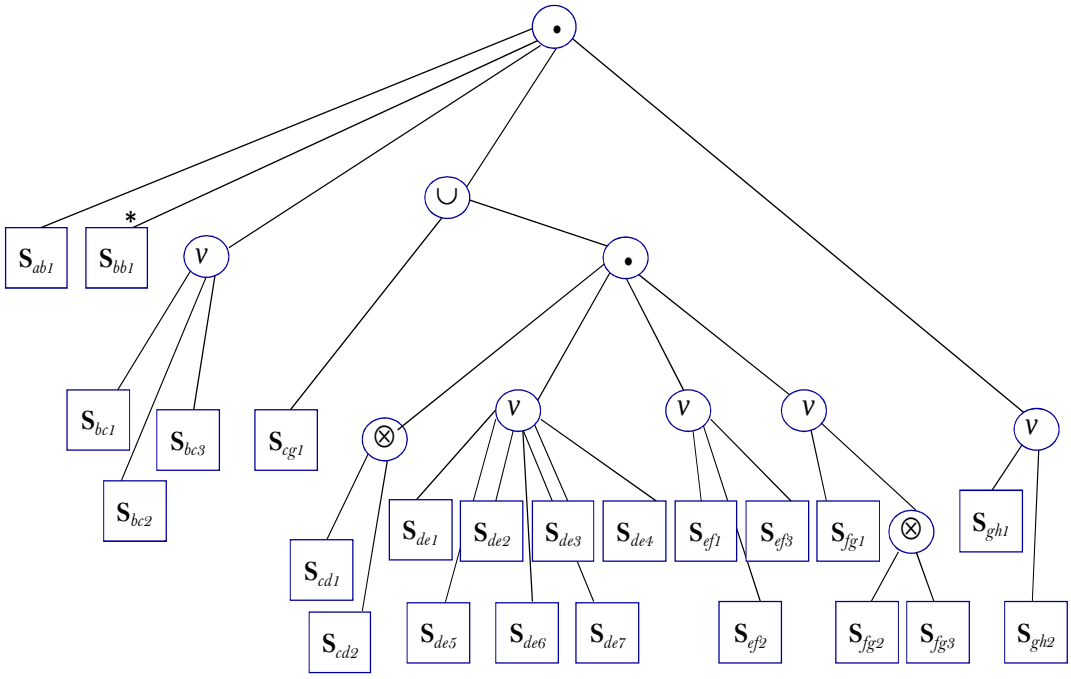

Figure 136. Représentation graphique du service $S_{Réaliser\ analyse\ plasma}$

10.3.2.2 *GENERER LES QOIS SIMPLE DE LA CARTE REALISER ANALYSE PLASMA*

La qualité d'un service intentionnel (*QoiS-Quality of intentional Service*) est définie dans le modèle *MiS-q* comme étant l'ensemble des buts qualité auxquels le service contribue à satisfaire partiellement (*cf.* Chapitre 6). La QoiS peut être simple lorsque le service est

atomique et elle est globale lorsque le service est agrégat. Il s'agit, dans ce qui suit, de générer les QoiS simple de la carte *Réaliser analyse plasma*.

La QoiS simple est associée au service atomique. Ce dernier, comme défini plus haut correspond à une section opérationnalisable de la carte des besoins (*cf.* Chapitre 4). La QoiS simple est alors générée à partir de l'ensemble des liens de contribution reliant la section opérationnalisable aux buts qualité (*cf.* Chapitre 6). Dans le modèle *MiS-q*, la QoiS simple est notée $\mathbf{S}_a.\{< q_j, d_j >\}$ (*cf.* Chapitre 6).

À partir de la Figure 135, nous générons alors la qualité offerte par les services atomiques (*i.e.* section opérationnalisable) de la carte *Réaliser analyse plasma*. Le Tableau 28 résume la QoiS simple fournie par les différents services atomiques du Tableau 26. Par exemple, les services atomiques $\mathbf{S}_{\text{Identifier tubes automatiquement}}$ (correspondant à la section bc_1), $\mathbf{S}_{\text{Identifier tubes par code à barre}}$ (correspondant à la section bc_2) et $\mathbf{S}_{\text{Identifier tubes manuellement}}$ (correspondant à la section bc_3) fournissent une QoiS correspondant, respectivement, à $\mathbf{S}_{\text{Identifier tubes automatiquement}}$.{<Précision,+>, <Temps,+>}, $\mathbf{S}_{\text{Identifier tubes par code à barre}}$.{< Précision,+>, <Temps,->} et $\mathbf{S}_{\text{Identifier tubes manuellement}}$.<Précision,->, <Temps,->}.

Tableau 28. Les QoiS simples de la carte *Réaliser analyse plasma*

Section	Service atomique	QoiS simple
ab1	$S_{\text{Authentifier l'opérateur}}$	$S_{\text{Authentifier l'opérateur}}$. {<Confidentialité, +>, <Temps, ->}
bc1	$S_{\text{Identifier tubes automatiquement}}$	$S_{\text{Identifier tubes automatiquement}}$. {<Précision, +>, <Temps, +>}
bc2	$S_{\text{Identifier tubes par code à barre}}$	$S_{\text{Identifier tubes par code à barre}}$. {<Précision, +>, <Temps,->}
bc3	$S_{\text{Identifier tubes manuellement}}$	$S_{\text{Identifier tubes manuellement}}$. {<Précision, ->,<Temps,->}
cg1	$S_{\text{Relancer dernière analyse}}$	$S_{\text{Relancer dernière analyse}}$.{<Précision,-->, <Temps, -->}
cd1	$S_{\text{Récupérer plasma par centrifugation}}$	$S_{\text{Récupérer plasma par centrifugation}}$. {<Précision, ->,<Temps,+>}
cd2	$S_{\text{Récupérer plasma par microfiltration}}$	$S_{\text{Récupérer plasma par microfiltration}}$. {<Précision, +>,<Temps,->}
ef1	$S_{\text{Analyser plasma sans incubation}}$	$S_{\text{Analyser plasma sans incubation}}$.{<Temps,+>}
ef2	$S_{\text{Analyser plasma avec une incubation}}$	$S_{\text{Analyser plasma avec une incubation}}$. {<Temps,->}
ef3	$S_{\text{Analyser plasma avec deux incubations}}$	$S_{\text{Analyser plasma avec deux incubation}}$. {<Temps,-->}
fg1	$S_{\text{Réaliser l'analyse par mesure chronométrique}}$	$S_{\text{Réaliser l'analyse par mesure chronométrique}}$. {<Temps, +>}
fg2	$S_{\text{Réaliser l'analyse par mesure immunologique}}$	$S_{\text{Réaliser l'analyse par mesure immunologique}}$. {<Temps, ->}
fg3	$S_{\text{Réaliser l'analyse par mesure colorimétrique}}$	$S_{\text{Réaliser l'analyse par mesure colorimétrique}}$. {<Temps, ->}
gh1	$S_{\text{Arrêter par validation résultat}}$	$S_{\text{Arrêter par validation résultat}}$. {<Précision, +>, <Temps, ->}
gh2	$S_{\text{Arrêter par complétude}}$	$S_{\text{Arrêter par complétude}}$. {<Précision,->, <Temps, +>}

10.3.3 ETAPE 3 : OPERATIONNALISATION DES SERVICES ATOMIQUES DE LA CARTE REALISER ANALYSE PLASMA

La troisième étape consiste à opérationnaliser les services atomiques obtenus précédemment (*cf.* Section 10.3.2.1.1). Cette opérationnalisation est effectuée suivant les sous-étapes suivantes (*cf.* Chapitre 7) :

5. Rechercher les services logiciels
6. Sélectionner un ou plusieurs services logiciels
7. Traduire les services logiciels
8. Agréger, si nécessaire, les services atomiques

10.3.3.1 *RECHERCHER LES SERVICES LOGICIELS*

Chaque service intentionnel atomique peut être opérationnalisé par un ou plusieurs services logiciels, de fournisseurs d'automates différents. Cette sous-étape permet de mettre en évidence les techniques permettant d'opérationnaliser un même service atomique. Par exemple, le service d'authentification de l'opérateur $S_{\text{Authentifier opérateur}}$ peut être opérationnalisé

par des services logiciels différents. En effet, le service d'authentification peut être implémenté par des techniques de mot de passe, de biométrie, de clé, *etc*. Ces techniques fournissent des QoS différentes.

10.3.3.2 *SELECTIONNER LES SERVICES LOGICIELS*

La société Diagnostica Stago choisit les services logiciels qu'elle souhaite implémenter. On suppose que la société a opté pour deux services logiciels, à savoir l'authentification par mot de passe et l'authentification par clé, comme techniques permettant d'opérationnaliser $\mathbf{S}_{\text{Authentifier opérateur}}$. La sélection de ces services est effectuée en prenant en compte leur QoS. Par ailleurs, la société choisit un seul service logiciel pour les autres services atomiques du processus d'analyse de plasma sanguin ($\mathbf{S}_{\text{Réaliser l'analyse par mesure chronométrique}}$, $\mathbf{S}_{\text{Réaliser l'analyse par mesure immunologique}}$, *etc.*).

10.3.3.3 *TRADUIRE LES SERVICES LOGICIELS*

Lorsque plusieurs services logiciels sont choisis par la société, pour opérationnaliser un même service atomique, ces derniers sont traduits en des services intentionnels atomiques, en augmentant les services logiciels d'une description intentionnelle. Par exemple, les services logiciels d'authentification par mot de passe et par clé sont augmentés d'une description intentionnelle, à savoir $\mathbf{S}_{\text{Authentifier par mot de passe}}$ et $\mathbf{S}_{\text{Authentifier par clé}}$. Par ailleurs, quand un seul service logiciel est choisi par la société, pour opérationnaliser un service atomique (*e.g.* $\mathbf{S}_{\text{Réaliser l'analyse par mesure chronométrique}}$, $\mathbf{S}_{\text{Réaliser l'analyse par mesure immunologique}}$, *etc.*), ce dernier garde son nom ou peut être augmenté du nom du service logiciel choisi.

De même, la qualité fournie par ces services logiciels est traduite en une QoiS simple, en consultant le référentiel qualité (*cf.* Chapitre 7), lequel permet de traduire la QoS (correspondant aux métriques) des services logiciels en des QoiS simple. Par exemple, la $\text{QoiS}(\mathbf{S}_{\text{Authentifier par mot de passe}}) = \mathbf{S}_{\text{Authentifier par mot de passe}}.\{<\text{Confidentialité},+>, <\text{Temps},->\}$, alors que $\text{QoiS}(\mathbf{S}_{\text{Authentifier par clé}}) = \mathbf{S}_{\text{Authentifier par clé}}.\{<\text{Confidentialité},++>, <\text{Temps},->\}$.

10.3.3.4 *AGREGER LES SERVICES ATOMIQUES*

Lorsque plusieurs services logiciels opérationnalisent un service atomique, ce dernier devient un service à choix alternatif (*cf.* Chapitre 7). Par exemple, le service atomique initial $\mathbf{S}_{\text{Authentifier opérateur}}$ devient un service à choix alternatif, qui agrège les différentes alternatives permettant de réaliser ce service, à savoir les services atomiques $\mathbf{S}_{\text{Authentifier par mot de passe}}$ et $\mathbf{S}_{\text{Authentifier par clé}}$. Ainsi, le service $\mathbf{S}_{\text{Authentifier opérateur}}$ devient le service à choix alternatif suivant : $\mathbf{S}_{\text{Authentifier opérateur}} = \otimes (\mathbf{S}_{\text{Authentifier par mot de passe}}, \mathbf{S}_{\text{Authentifier par clé}})$. Sa QoiS devient aussi une QoiS

globale, laquelle est définie comme suit : QoiS($\mathbf{S}_{\text{Authentifier opérateur}}$) = ⊗(QoiS($\mathbf{S}_{\text{Authentifier par mot de passe}}$), QoiS($\mathbf{S}_{\text{Authentifier par clé}}$)) (*cf.* Chapitre 6).

10.3.4 ETAPE 4 : GENERATION DE LA QOIS GLOBALE DE LA CARTE REALISER ANALYSE PLASMA

La quatrième étape permet de générer les QoiS globales de la carte *Réaliser analyse plasma*. Cette étape consiste à :

1. Générer la QoiS globale associée à chaque type de relation de la carte *Réaliser analyse plasma*
2. Générer la QoiS globale d'une section non opérationnalisable de la carte *Réaliser analyse plasma*

10.3.4.1 *GENERER LA QOIS GLOBALE ASSOCIEE A CHAQUE TYPE DE RELATION DE LA CARTE REALISER ANALYSE PLASMA*

La QoiS globale est associée au service agrégat. Le service agrégat, comme présenté précédemment, peut être un service à choix alternatif (noté par « ⊗ »), un service à choix multiple (noté par « *v* »), un service séquentiel (noté par « ● ») ou un service à variation de chemin (noté par « ∪ ») (*cf.* Chapitre 4). La QoiS globale correspond à ces différents types de service agrégat. La QoiS globale est alors dite une QoiS variante à choix alternatif (noté par « ⊗ »), une QoiS variante à choix multiple (noté par « *v* »), une QoiS à variation de chemin (noté par « ∪ »), ou une QoiS composite séquentielle (noté par « ● ») (*cf.* Chapitre 7).

Afin de générer la QoiS des différents chemins de la carte *Réaliser analyse plasma*, nous utilisons, comme défini dans le Chapitre 7, l'algorithme de MacNaughton et Yamada (MacNaugthon&Yamada,1960). En effet, cet algorithme est utilisé pour générer simultanément les différents services de la carte *Réaliser analyse plasma* et leur QoiS.

Le Tableau 29 résume les QoiS globale fournie par les différents services agrégats de la carte *Réaliser analyse plasma* ; ceux présentés au Tableau 27. Aussi, le Tableau 29 comporte la QoiS globale du service à choix alternatif $\mathbf{S}_{\text{Authentifier opérateur}}$ (*cf.* Section 10.3.3).

Tableau 29. Les QoiS globales de la carte *Réaliser analyse plasma*

QoiS globale	
QoiS variante à choix alternatif	QoiS($\mathbf{S}_{\text{Récupérer plasma}}$)=⊗(QoiS($\mathbf{S}_{\text{Récupérer plasma par centrifugation}}$),QoiS($\mathbf{S}_{\text{Récupérer plasma par microfiltration}}$))
	QoiS($\mathbf{S}_{\text{Mesurer plasma par photométrie}}$)=⊗(QoiS($\mathbf{S}_{\text{Mesurer plasma par immunologie}}$), QoiS($\mathbf{S}_{\text{Mesurer plasma par colorimétrie}}$))
	QoiS($\mathbf{S}_{\text{Authentifier opérateur}}$) = ⊗(QoiS($\mathbf{S}_{\text{Authentifier par mot de passe}}$), QoiS($\mathbf{S}_{\text{Authentifier par clé}}$))

QoiS variante à choix multiple	QoiS($\mathbf{S}_{\text{Identifier tubes}}$)= v(QoiS($\mathbf{S}_{\text{Identifier tubes automatiquement}}$), QoiS($\mathbf{S}_{\text{Identifier tubes par code à barre}}$),QoiS($\mathbf{S}_{\text{Identifier tubes manuellement}}$))
	QoiS($\mathbf{S}_{\text{Traiter plasma par choix test}}$)=$v$(QoiS($\mathbf{S}_{\text{Traiter plasma par TQ}}$), QoiS($\mathbf{S}_{\text{Traiter plasma par Fib}}$), QoiS($\mathbf{S}_{\text{Traiter plasma par VwF}}$), QoiS($\mathbf{S}_{\text{Traiter plasma par TCA}}$), QoiS($\mathbf{S}_{\text{Traiter plasma par ATIII}}$), QoiS($\mathbf{S}_{\text{Traiter plasma par TS}}$), QoiS($\mathbf{S}_{\text{Traiter plasma par TP}}$))
	QoiS($\mathbf{S}_{\text{Analyser plasma}}$)= v(QoiS($\mathbf{S}_{\text{Analyser plasma sans incubation}}$), QoiS($\mathbf{S}_{\text{Analyser plasma avec une incubation}}$),QoiS($\mathbf{S}_{\text{Analyser plasma avec deux incubations}}$))
	QoiS($\mathbf{S}_{\text{Mesurer plasma}}$)=$v$(QoiS($\mathbf{S}_{\text{Mesurer plasma par chronométrie}}$), QoiS($\mathbf{S}_{\text{Mesurer plasma par photométrie}}$))
	QoiS($\mathbf{S}_{\text{Arrêter analyse}}$)= v (QoiS($\mathbf{S}_{\text{Arrêter par validation résultat}}$), QoiS($\mathbf{S}_{\text{Arrêter par complétude}}$))
QoiS composite séquentielle	*QoiS* ($\mathbf{S}_{\text{Réaliser analyse plasma}}$)= ● (⊗(*QoiS*($\mathbf{S}_{ab1}$), *QoiS*($\mathbf{S}_{ab2}$)), *QoiS*($\mathbf{S}^*_{bb1}$), v (*QoiS*($\mathbf{S}_{bc1}$), *QoiS*($\mathbf{S}_{bc2}$), *QoiS*($\mathbf{S}_{bc3}$)), ∪(*QoiS*($\mathbf{S}_{cg1}$),●(⊗(*QoiS*($\mathbf{S}_{cd1}$), *QoiS*($\mathbf{S}_{cd2}$)), v (*QoiS*($\mathbf{S}_{de1}$), *QoiS*($\mathbf{S}_{de2}$), *QoiS*($\mathbf{S}_{de3}$), *QoiS*($\mathbf{S}_{de4}$), *QoiS*($\mathbf{S}_{de5}$), *QoiS*($\mathbf{S}_{de6}$), *QoiS*($\mathbf{S}_{de7}$)), v (*QoiS*($\mathbf{S}_{ef1}$), *QoiS*($\mathbf{S}_{ef2}$), *QoiS*($\mathbf{S}_{ef3}$)), v (*QoiS*($\mathbf{S}_{fg1}$), ⊗(*QoiS*($\mathbf{S}_{fg2}$), *QoiS*($\mathbf{S}_{fg3}$))))), v (*QoiS*($\mathbf{S}_{gh1}$), *QoiS*($\mathbf{S}_{gh2}$)))

Ainsi, la QoiS globale du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$ de la Figure 135 est définie par la formule suivante :

La QoiS globale du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, définie par la formule (*), devient alors :

QoiS ($\mathbf{S}_{\text{Réaliser analyse plasma}}$)= ● (⊗(*QoiS*($\mathbf{S}_{ab1}$), *QoiS*($\mathbf{S}_{ab2}$)), *QoiS*($\mathbf{S}^*_{bb1}$), v (*QoiS*($\mathbf{S}_{bc1}$), *QoiS*($\mathbf{S}_{bc2}$), *QoiS*($\mathbf{S}_{bc3}$)), ∪(*QoiS*($\mathbf{S}_{cg1}$),●(⊗(*QoiS*($\mathbf{S}_{cd1}$), *QoiS*($\mathbf{S}_{cd2}$)), v (*QoiS*($\mathbf{S}_{de1}$), *QoiS*($\mathbf{S}_{de2}$), *QoiS*($\mathbf{S}_{de3}$), *QoiS*($\mathbf{S}_{de4}$), *QoiS*($\mathbf{S}_{de5}$), *QoiS*($\mathbf{S}_{de6}$), *QoiS*($\mathbf{S}_{de7}$)), v (*QoiS*($\mathbf{S}_{ef1}$), *QoiS*($\mathbf{S}_{ef2}$), *QoiS*($\mathbf{S}_{ef3}$)), v (*QoiS*($\mathbf{S}_{fg1}$), ⊗(*QoiS*($\mathbf{S}_{fg2}$), *QoiS*($\mathbf{S}_{fg3}$))))), v (*QoiS*($\mathbf{S}_{gh1}$), *QoiS*($\mathbf{S}_{gh2}$))) (*)

La formule (*) indique que le service agrégat de la Figure 135 permet d'une part de réaliser le but *Réaliser analyse plasma*, et d'autre part de fournir une QoiS composite séquentielle. Cette QoiS correspond donc à la qualité du processus métier de l'analyse du plasma sanguin. Elle regroupe :

- la qualité fournie par le service d'authentification de l'opérateur (*i.e.* QoiS($\mathbf{S}_{\text{Authentifier opérateur}}$)) ;
- la qualité fournie par le service permettant d'effectuer le chargement des tubes (*i.e.* QoiS($\mathbf{S}_{\text{Charger les tubes}}$)) ;
- la qualité proposée par le service effectuant l'identification des tubes (*i.e.* QoiS($\mathbf{S}_{\text{Identifier les tubes}}$));

- la qualité exposée par le service réalisant la récupération du plasma (*i.e.* QoiS($\mathbf{S}_{\text{Récupérer plasma}}$)) ;
- la qualité offerte par les services qui traitent et analysent le plasma (*i.e.* QoiS($\mathbf{S}_{\text{Traiter plasma}}$) et QoiS($\mathbf{S}_{\text{Analyser plasma}}$), respectivement) ;
- la qualité fournie par le service qui réaliser la mesure du plasma (*i.e.* QoiS($\mathbf{S}_{\text{Mesurer plasma}}$)) ; et enfin
- la qualité des services permettant de finir le processus (*i.e.* QoiS($\mathbf{S}_{\text{Arrêter analyse}}$)).

La Figure 137 illustre la représentation graphique de la QoiS globale du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, à savoir la QoiS ($\mathbf{S}_{\text{Réaliser analyse plasma}}$). Comme le montre la Figure 137, la QoiS globale définie par la formule (*) permet de décrire la qualité globale du service réalisant le processus d'analyse de plasma sanguin (*i.e.* le service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$). Elle correspond à une composition de qualité, définie par la QoiS composite, noté « ● » et à une QoiS variante, introduite par les opérateurs de variation : « *v* », « ⊗ » ou « ∪». La QoiS variante implique qu'il est possible d'avoir différentes qualités, en fonction du service sélectionné et des exigences non-fonctionnelles des utilisateurs de l'automate.

10.3.4.2 *GENERER LA QOIS GLOBALE D'UNE SECTION NON-OPERATIONNALISABLE DE LA CARTE* $\boldsymbol{R}_{\text{EALISER ANALYSE PLASMA}}$

La QoiS d'une section non opérationnalisable de la carte *Réaliser analyse plasma* correspond à la QoiS globale de la carte qui l'affine. Comme présenté au Chapitre 4, la relation d'affinement permet de décrire une section non-opérationnalisable, à un niveau d'abstraction donné (*i*), par une carte à un niveau d'abstraction moins élevé (*i+1*). Il s'agit de générer la QoiS globale de la nouvelle carte. Le résultat obtenu au niveau (*i+1*) est alors transféré au niveau (*i*), suivant un raisonnement Botton-up.

Toutefois, toutes les sections de la carte *Réaliser analyse plasma* sont opérationnables. Pour plus de détail concernant la génération de la QoiS globale d'une section non-opérationnable, voir l'exemple présenté à la Section 7.6.2.

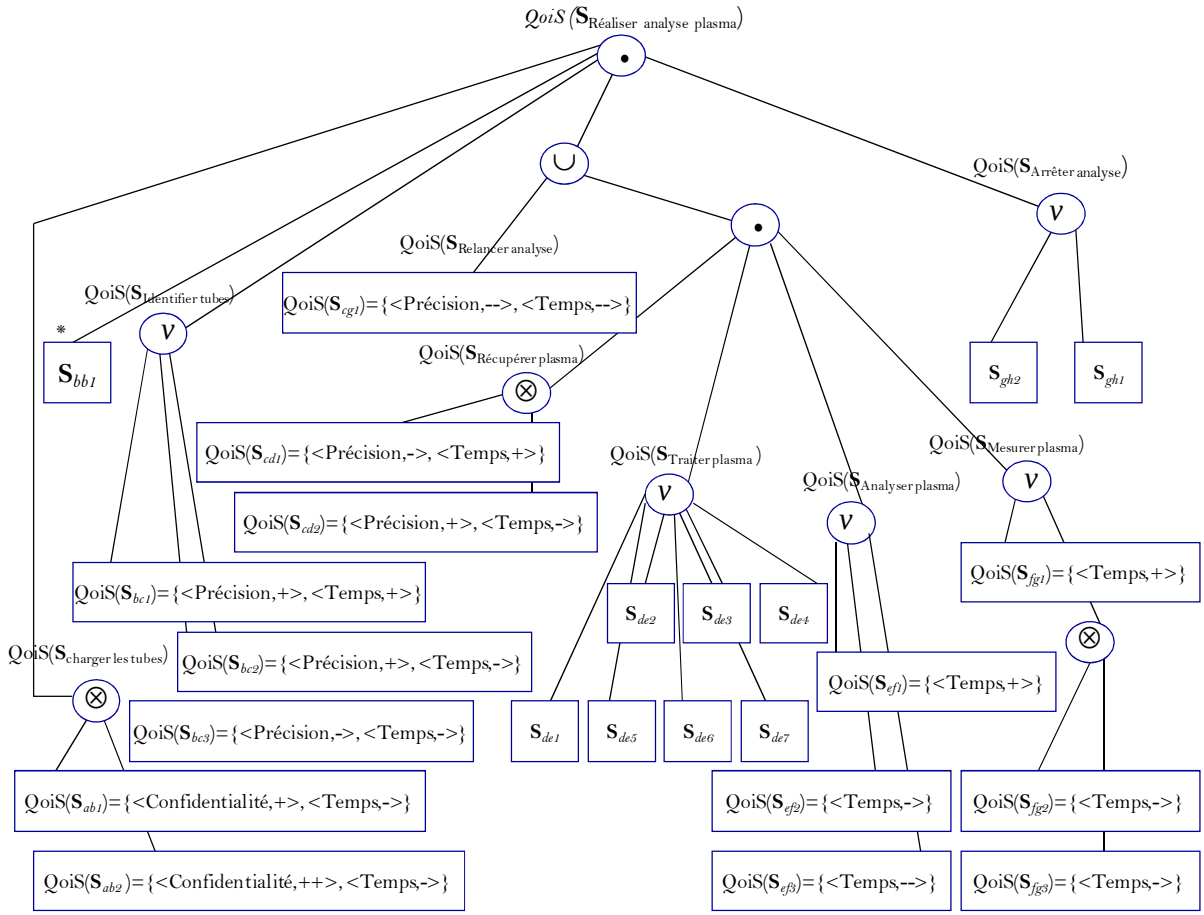

Figure 137. Représentation graphique de la QoiS ($\mathbf{S}_{\text{Réaliser analyse plasma}}$)

10.3.5 ETAPE 5 : ASSOCIATION DES CONTRAINTES DE DEPENDANCE AU SERVICE INTENTIONNEL AGREGAT $S_{REALISER\ ANALYSE\ PLASMA}$

La cinquième étape permet de considérer les contraintes de dépendance existantes entre les variantes du service agrégat $\mathbf{S}_{Réaliser\ analyse\ plasma}$. Cette étape consiste à :

1. Identifier les contraintes de dépendance du service agrégat $\mathbf{S}_{Réaliser\ analyse\ plasma}$
2. Décrire les contraintes de dépendance du service agrégat $\mathbf{S}_{Réaliser\ analyse\ plasma}$
3. Vérifier les règles de validité des contraintes de dépendance

10.3.5.1 *IDENTIFIER LES CONTRAINTES DE DEPENDANCE DU SERVICE AGREGAT $S_{REALISER\ ANALYSE\ PLASMA}$*

Le service intentionnel agrégat $\mathbf{S}_{Réaliser\ analyse\ plasma}$, via le service à variation, introduit de la variabilité dans la manière d'atteindre le but du service. Néanmoins, cette variabilité implique que toutes les combinaisons possibles ne sont pas forcément faisables ou acceptables. De manière similaire à une ligne de produits logiciels, nous avons proposé dans le modèle *MiS-q* (*cf.* Chapitre 6), d'associer au service intentionnel agrégat un ensemble de contraintes de dépendance. Ces contraintes sont définies entre les variantes de ce service. Celles-ci, dans le

contexte d'un service agrégat, peuvent être des contraintes qui expriment des contraintes d'inclusion (contrainte de type Exige) ou d'exclusion (contrainte de type Exclut). Ces contraintes permettent de valider l'exécutabilité des configurations du service agrégat (*cf.* Chapitre 6), et donc sa cohérence.

Nous avons identifié un ensemble de contraintes associées au service agrégat $\mathbf{S}_{Réaliser\ analyse\ plasma}$. Ces contraintes sont identifiées notamment à partir de l'analyse du processus métier de l'analyse du plasma sanguin. Celles-ci sont également identifiées à partir des contraintes techniques produites par les services logiciels, lesquels opérationnalisent les services atomiques.

En effet, le processus métier de l'analyse du plasma sanguin exige que lorsque l'automate effectue un chargement urgent des tubes, il doit également permettre d'effectuer une identification manuelle de ces tubes.

Ce processus métier précise que l'automate qui effectue des tests de TS et des tests de TP, écarte l'utilisation de méthode avec incubation.

L'utilisation de la méthodologie sans incubation empêche l'automate de mesurer le plasma avec une mesure photométrique.

Aussi, l'assignation des tests de ATIII et des tests de TCA nécessite une analyse avec une seule incubation du plasma. De plus, Le test ATIII requiert la méthode de mesure immunologique, interdisant ainsi d'appliquer les méthodes colorimétrique et chronométrique. Le test TCA, quant à lui, nécessite d'être mesuré par la méthode colorimétrique.

Egalement, le processus métier de l'analyse du plasma sanguin exige que l'utilisation des tests TQ, Fib, VwF, nécessite une analyse avec deux incubations du plasma. Enfin, le test TQ requiert la méthode de mesure colorimétrique, alors que le test Fib nécessite d'être mesuré par la méthode chronométrique. Le test VwF a besoin de la mesure immunologique.

10.3.5.2 *DECRIRE LES CONTRAINTES DE DEPENDANCE DU SERVICE AGREGAT* $S_{REALISER\ ANALYSE\ PLASMA}$

La description des contraintes consiste à assimiler les contraintes de dépendance Exige et Exclut à la logique booléenne, telle que définie dans *MiS-q* (*cf.* Chapitre 6). La contrainte Exige est assimilé à l'opérateur d'implication ($\Rightarrow$), tandis que la contrainte Exclut est assimilé aux opérateurs de négation et d'implication ($\Rightarrow \neg$).

Le Tableau 30 résume la description des contraintes de dépendance, identifiées ci-dessus (*cf.* Section 10.3.5.1), lesquelles sont associées au service intentionnel agrégat $\mathbf{S}_{Réaliser\ analyse}$

plasma. Par exemple, la contrainte d'exclusion qui précise que l'utilisation de la méthodologie sans incubation exclut de mesurer le plasma avec une mesure photométrique, est décrite comme suit : $\mathbf{S}_{Analyser\ plasma\ sans\ incubation} \Rightarrow \neg(\mathbf{S}_{Mesurer\ plasma\ avec\ photométrie})$. Aussi, la contrainte d'exigence qui définit que l'assignation d'un test ATIII nécessite une analyse avec une seule incubation du plasma, est décrite comme suit : $\mathbf{S}_{Traiter\ plasma\ par\ ATIII} \Rightarrow \mathbf{S}_{Analyser\ plasma\ avec\ une\ incubation}$. La

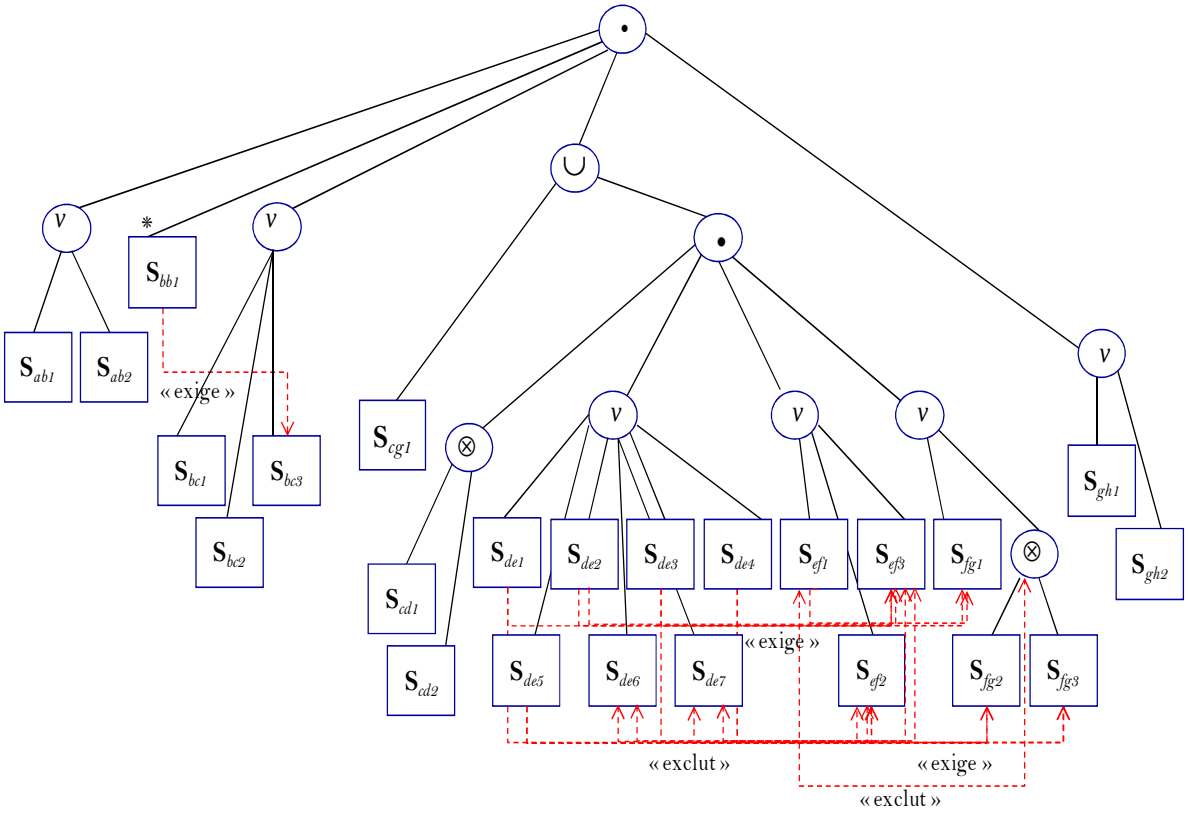

Figure 138 met en évidence la représentation graphique de l'ensemble de ces contraintes. Celles-ci doivent être satisfaites par les configurations exécutables du service agrégat $\mathbf{S}_{Réaliser\ analyse\ plasma}$.

Tableau 30. Les contraintes de dépendance de $\mathbf{S}_{Réaliser\ analyse\ plasma}$

Services	Contrainte de dépendance
$\mathbf{S}_{Charger\ tube\ par\ mode\ urgent}$	$\mathbf{S}_{Charger\ tube\ par\ mode\ urgent} \Rightarrow \mathbf{S}_{Identifier\ tube\ manuellement}$
$\mathbf{S}_{Traiter\ plasma\ par\ TS}$	$\mathbf{S}_{Traiter\ plasma\ par\ TS} \Rightarrow \neg\ \mathbf{S}_{Analyser\ plasma\ avec\ une\ incubation}$ $\mathbf{S}_{Traiter\ plasma\ par\ TS} \Rightarrow \neg\ \mathbf{S}_{Analyser\ plasma\ avec\ deux\ incubations}$
$\mathbf{S}_{Traiter\ plasma\ par\ TP}$	$\mathbf{S}_{Traiter\ plasma\ par\ TP} \Rightarrow \neg\ \mathbf{S}_{Analyser\ plasma\ avec\ une\ incubation}$ $\mathbf{S}_{Traiter\ plasma\ par\ TP} \Rightarrow \neg\ \mathbf{S}_{Analyser\ plasma\ avec\ deux\ incubations}$
$\mathbf{S}_{Analyser\ plasma\ sans\ incubation}$	$\mathbf{S}_{Analyser\ plasma\ sans\ incubation} \Rightarrow \neg(\mathbf{S}_{Mesurer\ plasma\ avec\ photométrie})$
$\mathbf{S}_{Traiter\ plasma\ par\ ATIII}$	$\mathbf{S}_{Traiter\ plasma\ par\ ATIII} \Rightarrow \mathbf{S}_{Analyser\ plasma\ avec\ une\ incubation}$

	$S_{Traiter\ plasma\ par\ ATIII} \Rightarrow S_{Mesurer\ plasma\ avec\ immunologie}$
$S_{Traiter\ plasma\ par\ TCA}$	$S_{Traiter\ plasma\ par\ TCA} \Rightarrow S_{Analyser\ plasma\ avec\ une\ incubation}$
	$S_{Traiter\ plasma\ par\ TCA} \Rightarrow S_{Mesurer\ plasma\ avec\ colorimétrie}$
$S_{Traiter\ plasma\ par\ TQ}$	$S_{Traiter\ plasma\ par\ TQ} \Rightarrow S_{Analyser\ plasma\ avec\ deux\ incubations}$
	$S_{Traiter\ plasma\ par\ TQ} \Rightarrow S_{Mesurer\ plasma\ avec\ colorimétrie}$
$S_{Traiter\ plasma\ par\ Fib}$	$S_{Traiter\ plasma\ par\ Fib} \Rightarrow S_{Analyser\ plasma\ avec\ deux\ incubation}$
	$S_{Traiter\ plasma\ par\ Fib} \Rightarrow S_{Mesurer\ plasma\ avec\ chronométrie}$
$S_{Traiter\ plasma\ par\ VwF}$	$S_{Traiter\ plasma\ par\ VwF} \Rightarrow S_{Analyser\ plasma\ avec\ deux\ incubation}$
	$S_{Traiter\ plasma\ par\ VwF} \Rightarrow S_{Mesurer\ plasma\ avec\ immunologie}$

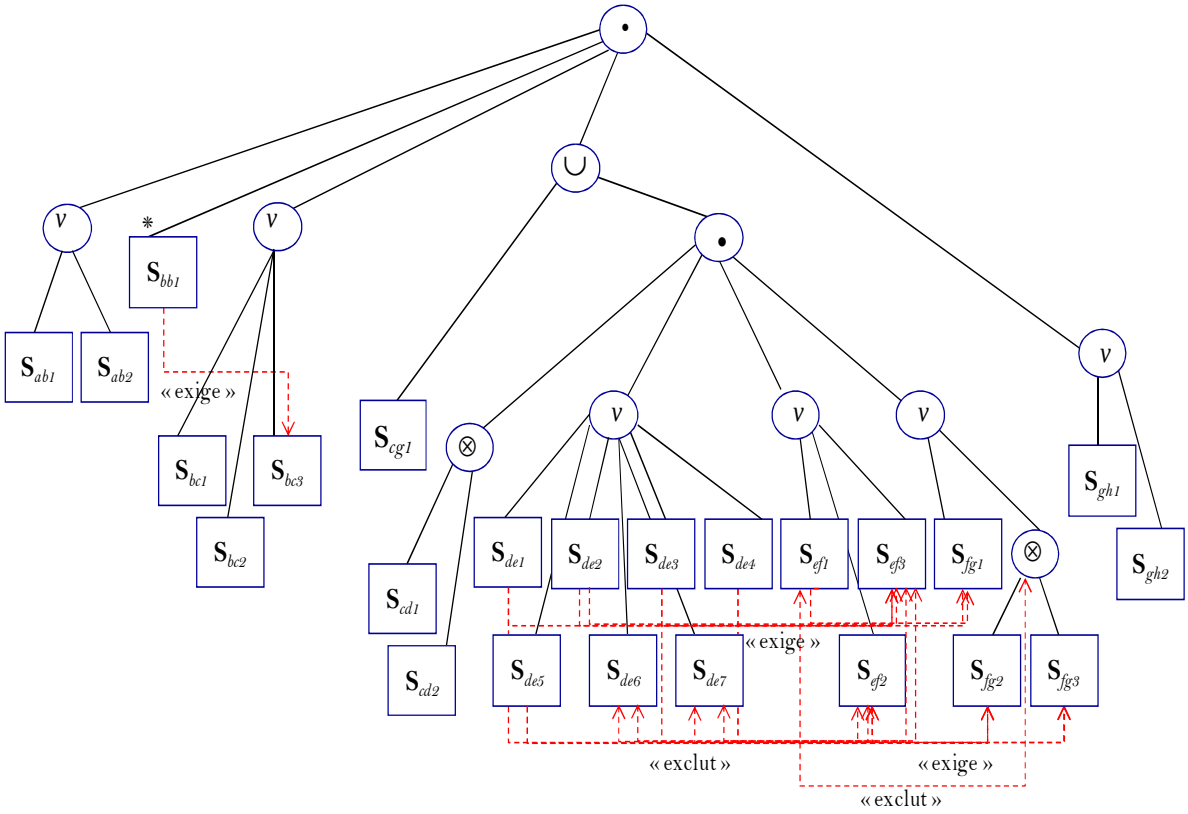

Figure 138. Représentation graphique des contraintes de dépendance de $S_{Réaliser\ analyse\ plasma}$

10.3.5.3 *VERIFIER LES REGLES DE COHERENCE DES CONTRAINTES DE DEPENDANCE*

La vérification des règles de cohérence consiste à vérifier en premier lieu que les contraintes comprennent une seule cible. Si plusieurs cibles sont introduites alors les contraintes sont réécrites en autant de contraintes de façon à ce que chacune d'entre elles ne contienne qu'une seule cible. Ensuite, il s'agit de vérifier que les contraintes ne transgressent pas la sémantique des opérateurs d'agrégation.

Le prototype TOPSiQC (*cf.* Chapitre 9) permet de générer et de vérifier l'exécutabilité des configurations de ce service agrégat. Ensuite, il les compare en vue de sélectionner celles qui

répondent au mieux au contexte qualité de l'opérateur. L'utilisation de TOPSiQC dans l'exemple du service agrégat $\mathbf{S}_{\textit{Réaliser analyse plasma}}$ est décrit dans la section suivante.

10.4 *APPLICATION DU PROTOTYPE TOPSIQC AU SERVICE $S_{REALISER\ ANALYSE\ PLASMA}$*

Dans le cas du laboratoire d'analyse que nous avons présenté précédemment (*cf.* Section 10.2.3), le prototype TOPSiQC (*cf.* Chapitre 9) permet de guider les opérateurs à sélectionner, parmi les divers automates du laboratoire, ceux qui pourraient répondre au mieux à leurs exigences non-fonctionnelles, relatives à une demande d'analyse particulière. Notamment, le prototype TOPSiQC permet de guider les opérateurs à configurer le service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$, afin de sélectionner les configurations de $\mathbf{S}_{\text{Réaliser analyse plasma}}$ qui pourraient répondre au mieux à ces exigences non-fonctionnelles.

Nous avons décidé dans la suite de ce chapitre de ne présenter que la configuration du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$, en fonction des exigences non-fonctionnelles des opérateurs. Comme défini dans le Chapitre 8, les exigences non-fonctionnelles sont formalisées dans un contexte qualité, nommé *Cq*. Ce dernier peut être composé de plusieurs qualités à satisfaire. Le service $\mathbf{S}_{\text{Réaliser analyse plasma}}$ contient plusieurs points de variation pour lesquels il faut à chaque fois sélectionner les variantes les plus adéquates pour satisfaire le contexte qualité de l'agent métier. En faisant la configuration du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, on montre également le processus de sélection, car le processus de configuration consiste à choisir parmi les différentes configurations de ce service celles qui répondent au mieux au contexte qualité.

Ainsi, nous utilisons le prototype TOPSiQC (*cf.* Chapitre 9) afin de configurer le service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$, de sorte à trouver, lors d'une demande d'analyse, toutes les configurations de ce service répondant au mieux au contextes qualité des opérateurs. Nous proposons, dans ce qui suit, de détailler les différentes étapes permettant au protocole TOPSiQC de configurer le service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$. Ces étapes se décline en :

1. Etape 1 : définition des données d'entrée de TOPSiQC
2. Etape 2 : définition des traitements de TOPSiQC
3. Etape 3 : définition des résultats de TOPSiQC

10.4.1 ETAPE1 : DEFINITION DES DONNEES D'ENTREE DE TOPSIQC

Le prototype TOPSiQC utilise trois fichiers principaux comme paramètres d'entrée (*cf.* Chapitre 9). Ces fichiers permettent :

1. La spécification du contexte qualité représentant les exigences des opérateurs
2. La spécification du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$
3. La spécification des contraintes de dépendance associées au service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$

10.4.1.1 *LA SPECIFICATION DU CONTEXTE QUALITE*

Nous avons décidé de prendre cinq contextes qualité différents afin de montrer l'importance de la configuration intentionnelle. En effet, les différentes variantes du service intentionnel agrégat peuvent plus ou moins satisfaire des qualités. Nous avons choisi de présenter plusieurs contextes qualité afin de montrer les répercussions du contexte qualité sur le processus de configuration et le choix des variantes.

Les Figure 139, Figure 140, Figure 141, Figure 142 et Figure 143 montrent cinq exemples différents de contexte qualité (nommé *Cqi/i=1..5*) dont la description se base sur celle définie à la Section 9.3.1.1. En effet, lorsque l'opérateur (ou le biologiste ou le laborantin) désire réaliser une analyse de sang, il souhaite également satisfaire des qualités particulières aux conditions d'analyse du moment. Les cinq contextes représentent des situations différentes d'analyse : les deux premiers contextes (*Cq1* et *Cq2*) consistent à satisfaire une seule qualité à la fois, les deux suivants admettent de satisfaire deux qualités contradictoires (avec précision des priorités), et enfin le dernier contexte correspond à satisfaire trois qualités simultanément. Ces qualités sont définies dans les différents contextes suivants :

Pour *Cq1* (*cf.* Figure 139) : l'opérateur souhaite satisfaire la Précision (lignes 9 à la Figure 139) avec un seuil minimal de satisfaction égale à « + » (lignes 10 à la Figure 139).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <context:Context xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4        xmlns:context='http://LaboratoireAnalyse/Context'
5        xsi:schemaLocation='http://LaboratoireAnalyse/Context
6  Context.xsd'>
7
8  <context:QoS>
9      <context:Quality>Précision</context:Quality>
10     <context:Degree>+</context:Degree>
11     <context:Priority>1</context:Priority>
12 </context:QoS>
13
14 </context:Context>
```

Figure 139. Le contexte qualité *Cq1*

Pour *Cq2* (*cf.* Figure 140) : l'opérateur souhaite satisfaire la Confidentialité (ligne 8 à la Figure 140) avec un seuil minimal de satisfaction conforme à « + » (lignes 9 à la Figure 140).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
```

```
3   <context:Context xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4         xmlns:context='http://LaboratoireAnalyse/Context'
5         xsi:schemaLocation='http://LaboratoireAnalyse/Context
6   Context.xsd'>
7   <context:QoS>
8       <context:Quality>Confidentialité</context:Quality>
9       <context:Degree>+</context:Degree>
10      <context:Priority>1</context:Priority>
11  </context:QoS>
12
    </context:Context>
```

Figure 140. Le contexte qualité *Cq2*

Pour *Cq3* (*cf.* Figure 141) : l'opérateur souhaite satisfaire deux qualités, à savoir la Confidentialité (ligne 9 à la Figure 141) et la Précision (ligne 14 à la Figure 141), avec un seuil égale à « + » (lignes 10 et 15 à la Figure 141) pour chacune de ces qualités. L'opérateur privilégie la Précision (*Priorité=0,7*) (lignes 16 à la Figure 141) à la Confidentialité (*Priorité=0,3*) (lignes 11 à la Figure 141).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2
3   <context:Context xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4         xmlns:context='http://LaboratoireAnalyse/Context'
5         xsi:schemaLocation='http://LaboratoireAnalyse/Context
6   Context.xsd'>
7
8   <context:QoS>
9       <context:Quality>Confidentialité</context:Quality>
10      <context:Degree>+</context:Degree>
11      <context:Priority>0.3</context:Priority>
12  </context:QoS>
13  <context:QoS>
14      <context:Quality>Précision</context:Quality>
15      <context:Degree>+</context:Degree>
16      <context:Priority>0.7</context:Priority>
17  </context:QoS>
18
    </context:Context>
```

Figure 141. Le contexte qualité *Cq3*

Pour *Cq4* (celui de la Figure 142) : l'opérateur souhaite satisfaire la Précision (ligne 9 à la Figure 142) et le Temps (ligne 14 à la Figure 142) avec un seuil de satisfaction minimal égale à « + » (lignes 10 et 15 à la Figure 142) pour chacune des qualités. L'opérateur privilégie le Temps (*Priorité=0,7*) (lignes 16 à la Figure 142 à la Précision (*Priorité=0,3*) (lignes 11 à la Figure 142).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2
3   <context:Context xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4         xmlns:context='http://LaboratoireAnalyse/Context'
5         xsi:schemaLocation='http://LaboratoireAnalyse/Context
6   Context.xsd'>
7
8   <context:QoS>
```

```
9       <context:Quality>Précision</context:Quality>
10      <context:Degree>+</context:Degree>
11      <context:Priority>0.3</context:Priority>
12  </context:QoS>
13  <context:QoS>
14      <context:Quality>Temps</context:Quality>
15      <context:Degree>+</context:Degree>
16      <context:Priority>0.7</context:Priority>
17  </context:QoS>
18
    </context:Context>
```

Figure 142. Le contexte qualité *Cq4*

Pour *Cq5* (celui de la Figure 143) : l'opérateur souhaite satisfaire toutes les qualités, à savoir la Précision (ligne 9 à la Figure 143), le Temps (ligne 14 à la Figure 143) et la Confidentialité (ligne 19 à la Figure 143) avec un seuil de satisfaction minimal égale à « + » (lignes 10, 15 et 20 à la Figure 143) pour chacune de ces qualités. L'opérateur privilégie le Temps, avec une priorité égale à « *0,4* » (lignes 16 à la Figure 143), à la Précision et à la Confidentialité, auxquelles il a affecté la priorité de « *0,3* » (lignes 11 et 21 à la Figure 143).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2
3   <context:Context xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4         xmlns:context='http://LaboratoireAnalyse/Context'
5         xsi:schemaLocation='http://LaboratoireAnalyse/Context
6   Context.xsd'>
7
8   <context:QoS>
9       <context:Quality>Précision</context:Quality>
10      <context:Degree>+</context:Degree>
11      <context:Priority>0.3</context:Priority>
12  </context:QoS>
13  <context:QoS>
14      <context:Quality>Temps</context:Quality>
15      <context:Degree>+</context:Degree>
16      <context:Priority>0.4</context:Priority>
17  </context:QoS>
18  <context:QoS>
19      <context:Quality>Confidentialité</context:Quality>
20      <context:Degree>+</context:Degree>
21      <context:Priority>0.3</context:Priority>
22  </context:QoS>
23
    </context:Context>
```

Figure 143. Le contexte qualité *Cq5*

10.4.1.2 *LA SPECIFICATION DU SERVICE $S_{REALISER\ ANALYSE\ PLASMA}$*

La formule du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$ est définie comme suit (*cf.* Section 10.3.2.1.2) : $\mathbf{S}_{\text{Réaliser analyse plasma}} = \bullet\ (\otimes(\mathbf{S}_{ab1}, \mathbf{S}_{ab2}), \mathbf{S^*}_{bb1}, \nu\ (\mathbf{S}_{bc1},\mathbf{S}_{bc2}, \mathbf{S}_{bc3}), \cup(\mathbf{S}_{cg1},\bullet(\otimes\ (\mathbf{S}_{cd1}, \mathbf{S}_{cd2}), \nu\ (\mathbf{S}_{de1},\mathbf{S}_{de2}, \mathbf{S}_{de3},\mathbf{S}_{de4},\mathbf{S}_{de5},\mathbf{S}_{de6},\mathbf{S}_{de7}), \nu\ (\mathbf{S}_{ef1},\mathbf{S}_{ef2}, \mathbf{S}_{ef3}), \nu\ (\mathbf{S}_{fg1}, \otimes\ (\mathbf{S}_{fg2}, \mathbf{S}_{fg3})))), \nu(\mathbf{S}_{gh1}, \mathbf{S}_{gh2}))$.

Le service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$, correspondant à la carte de la Figure 135, est décrit sous forme d'un arbre dans lequel les opérateurs d'agrégation « ● », « ∪», « ∨ », «⊗ » et « * » sont remplacés, respectivement, par *And*, *Xor*, *Or*, *LXor*, et *Star*. La Figure 144 montre une partie de la description XML du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, la description complète de ce service se trouve à l'Annexe 2. Dans cette description, nous retrouvons notamment les qualités offertes par chaque service atomique participant à la composition. Par exemple, le service $\mathbf{S}_{ab1}$ (ligne 11 à la Figure 144), correspondant au service $\mathbf{S}_{\text{Authentifier l'opérateur par mot de passe}}$, contribue à la satisfaction des qualités de Confidentialité (ligne 13 à la Figure 144) et de Temps (ligne 18 à la Figure 144) avec les seuils de satisfaction, respectifs, « + » (ligne 15 à la Figure 144) et « + » (ligne 19 à la Figure 144).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2
3   <tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
4         xmlns:tree='http://DiagnosticaStago/Tree'
5         xsi:schemaLocation='http://DiagnosticaStago/Tree Tree.xsd'>
6
7   <tree:And>
8
9       <tree:LXor>
10        <tree:Leaf>
11          <tree:Value>Sab1 </tree:Value>
12                      <tree:QoS>
13                          <tree:Quality>Confidentialité
14                          </tree:Quality>
15                          <tree:Degree>+</tree:Degree>
16                      </tree:QoS>
17                      <tree:QoS>
18                          <tree:Quality>Temps</tree:Quality>
19                          <tree:Degree>+</tree:Degree>
20                      </tree:QoS>
21        </tree:Leaf>
22        <tree:Leaf>
23          <tree:Value>Sab2 </tree:Value>
24                       <tree:QoS>
25                           <tree:Quality>Confidentialité
26                           </tree:Quality>
27                           <tree:Degree>++</tree:Degree>
28                       </tree:QoS>
29                       <tree:QoS>
30                           <tree:Quality>Temps</tree:Quality>
31                           <tree:Degree>-</tree:Degree>
32                       </tree:QoS>
33        </tree:Leaf>
34     </tree:LXor>
35       ……..
36
37  </tree:And>
38  </tree:Services>
```

Figure 144. La spécification du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$

10.4.1.3 *LA SPECIFICATION DES CONTRAINTES DE DEPENDANCE DU SERVICE* $S_{REALISER\ ANALYSE\ PLASMA}$

Les contraintes de dépendance existantes entre les différentes variantes du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$ ont été présenté ci-dessus (*cf.* Section 10.3.2.1.2), ainsi que leur représentation graphique (*cf.* Figure 135).

La Figure 145 montre la description XML des contraintes de dépendance du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$. Ces contraintes sont valides dans le contexte du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$ (ligne 9 à la Figure 145). Par exemple, le service $\mathbf{S}_{bb1}$ (*i.e.* $\mathbf{S}_{\text{Charger tube par mode urgent}}$) exige (*Imply* à la Figure 145) le service $\mathbf{S}_{bc3}$ (*i.e.* $\mathbf{S}_{\text{Identifier tube manuellement}}$) (ligne 11 à la Figure 145), tandis que le service $\mathbf{S}_{ef1}$ (*i.e.* $\mathbf{S}_{\textit{Analyser plasma sans incubation}}$) exclut (*Exclude* à la Figure 145) le service *lxor-03* (*i.e.* $\mathbf{S}_{\textit{Mesurer plasma avec photométrie}}$) (ligne 16 à la Figure 145), qui correspond au service à choix alternatif $\otimes(\mathbf{S}_{fg2},\mathbf{S}_{fg3})$ (voir descriptif du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$ à l'Annexe 2).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <dependence:Dependence xmlns:xsi='http://www.w3.org/2001/XMLSchema-
4  instance'
5        xmlns:dependence='http://xml.netbeans.org/schema/Dependence'
6        xsi:schemaLocation='http://xml.netbeans.org/schema/Dependence
7  Dependence.xsd'>
8
9  <Service agrégat> ref= "S Réaliser analyse plasma" </Service agrégat>
10
11 <dependence:Imply source="Sbb1" target="Sbc3" />
12 <dependence:Exclude source="Sde6" target="Sef2" />
13 <dependence:Exclude source="Sde6" target="Sef3" />
14 <dependence:Exclude source="Sde7" target="Sef2" />
15 <dependence:Exclude source="Sde7" target="Sef3" />
16 <dependence:Exclude source="Sef1" target="lxor-03" />
17 <dependence:Imply source="Sde5" target="Sef2" />
18 <dependence:Imply source="Sde5" target="Sfg2" />
19 <dependence:Imply source="Sde4" target="Sef2" />
20 <dependence:Imply source="Sde4" target="Sfg3" />
21 <dependence:Imply source="Sde1" target="Sef3" />
22 <dependence:Imply source="Sde1" target="Sfg3" />
23 <dependence:Imply source="Sde2" target="Sef3" />
24 <dependence:Imply source="Sde2" target="Sfg1" />
25 <dependence:Imply source="Sde3" target="Sef3" />
26 <dependence:Imply source="Sde3" target="Sfg2" />
27
28 </dependence:Dependence>
```

Figure 145. La spécification des contraintes de dépendance du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$

10.4.2 ETAPE 2 : DEFINITION DES TRAITEMENTS DE TOPSIQC

Une fois les fichiers d'entrée (contextes (*cf.* Section 10.4.1.1), service (*cf.* Section 10.4.1.2) et contrainte (*cf.* Section 10.4.1.3) approuvés (par rapport aux descriptions XSD), le prototype TOPSiQC (*cf.* Chapitre 9) exécute un certain nombre de traitements. Ces traitements reflètent l'adaptation des différentes étapes de la méthode TOPSIS au cas de la configuration

intentionnelle des services (*cf.* Chapitre 8). Ces traitements se décline en plusieurs étapes, tels que présenté plus haut (*cf.* Chapitre 8 et Chapitre 9) :

1. Prétraitement du prototype TOPSiQC
2. Construction de la matrice pondérée
3. Définition des alternatives artificielles
4. Calcul des distances euclidiennes
5. Calcul de la proximité (ou la similarité) relative.

10.4.2.1 *PRETRAITEMENT DU PROTOTYPE TOPSIQC*

La première étape consiste à définir les différentes configurations exécutables à comparer, les buts qualité qui correspondent aux critères de sélection, et les $QoiS_{moy}$ de ces configurations.

GENERER LES CONFIGURATIONS DU SERVICE $S_{REALISER\ ANALYSE\ PLASMA}$

Le prototype TOPSiQC génère les différentes configurations du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, à partir de la description présentée plus haut (*cf.* Figure 144). La génération des configurations dépend des opérateurs d'agrégation « $\cup$ », « $\vee$ », « $\otimes$ » et « * » (*cf.* Chapitre 8). Chaque chemin dans le service agrégat représente une configuration possible de service. Pour le service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, TOPSiQC génère *3476* configurations exécutables ; *i* représente le numéro de la configuration. Cette dernière est désignée par le code $\boldsymbol{Conf}_{i/i=1..m}$.

VALIDER LES CONFIGURATIONS DU SERVICE $S_{REALISER\ ANALYSE\ PLASMA}$

Le prototype TOPSiQC valide l'exécutabilité des configurations, en fonction des contraintes de dépendance (*cf.* Figure 146) définies entre les variantes du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$. En effet, TOPSiQC vérifie si tous les services composants d'une configuration sont compatibles, c'est-à-dire que leur composition respecte le processus métier d'analyse de plasma sanguin, ainsi que les contraintes de dépendance définies sur ce processus. Les configurations qui ne sont pas exécutables sont éliminées du processus de sélection, alors que les configurations exécutables sont gardées et comparées suivant les différents contextes qualité.

CALCULER LA $QOIS_{MOY}$ DES CONFIGURATIONS DU SERVICE $S_{REALISER\ ANALYSE\ PLASMA}$

Le prototype TOPSiQC calcule la qualité moyenne de chaque configuration exécutable. Il considère les qualités que l'opérateur souhaite satisfaire, à savoir la Précision pour *Cq1*, la Confidentialité pour *Cq2*, la Confidentialité et la Précision pour *Cq3*, le Temps et la Précision

pour *Cq4* et le Temps, la Précision et la Confidentialité pour *Cq5*, et calcule, pour ces qualités, la $QoiS_{moy}$ de chaque configuration exécutable.

En effet, le prototype TOPSiQC récupère, pour chaque contexte qualité *Cq1*, *Cq2, Cq3, Cq4* et *Cq5* (*cf.* Section 10.4.1.1), les buts qualité que l'opérateur souhaite satisfaire. Le prototype TOPSiQC calcule, pour chaque but qualité du contexte, la $QoiS_{moy}$ de chaque configuration exécutable, en fonction de la QoiS simple des services atomiques. Celle-ci est définie dans la description du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$ (*cf.* Annexe 2). La valeur de la QoiS simple peut être « ++ » (Très satisfait), « + » (Satisfait), « - » (Peu satisfait) et « -- » (Pas du tout satisfait). Lorsque le service n'a aucune contribution à un but qualité donné (celui qui est défini dans le contexte qualité), TOPSiQC lui affecte le seuil « *?* » (Neutre). Le prototype TOPSiQC se base sur des valeurs numériques pour calculer la $QoiS_{moy}$. Pour cela, les seuils de satisfaction sont normalisés en leur affectant des valeurs numériques : « ++ » est remplacé par « *2* », « ++ » par « *1* », *etc*. (*cf.* Tableau 24). Ces valeurs sont alors utilisées pour calculer la qualité moyenne de la configuration exécutable (*cf.* Chapitre 6). Celle-ci est calculée dynamiquement, chaque fois que l'opérateur fournit son contexte qualité. L'utilisation de la qualité moyenne au lieu de la qualité minimale est justifiée par ce qui suit. Nous considérons les deux configurations suivantes : $\mathbf{Conf}_1=\bullet(\mathbf{S}_{ab1},\mathbf{S}_{bc3},\mathbf{S}_{cd1},\mathbf{S}_{de6},\mathbf{S}_{ef1},\mathbf{S}_{fg2},\mathbf{S}_{gh1})$ et $\mathbf{Conf}_2=\bullet(\mathbf{S}_{ab1},\mathbf{S}_{bc2},\mathbf{S}_{cd2},\mathbf{S}_{de6},\mathbf{S}_{ef1},\mathbf{S}_{fg2},\mathbf{S}_{gh2})$. Concernant la précision, la $QoiS_{min}(\mathbf{Conf}_1)$ = $QoiS_{min}(\mathbf{Conf}_2)$ = {<Précision,->}. Les deux configurations fournissent le même seuil, alors que la configuration $\mathbf{Conf}_1$ fournit un seul « + » grâce au service $\mathbf{S}_{gh1}$ et plusieurs « - », et la configuration $\mathbf{Conf}_2$ offre un seul « - » via le service $\mathbf{S}_{gh2}$ et plusieurs « + ». Contrairement à la $QoiS_{min}$, la $QoiS_{moy}$ fournit une qualité plus objective, à savoir : la $QoiS_{moy}(\mathbf{Conf}_1)$= {<Précision,-1/3>} et $QoiS_{moy}(\mathbf{Conf}_2)$={<Précision,1/3>}. La $\mathbf{Conf}_2$ fournit alors une meilleure qualité que $\mathbf{Conf}_1$, concernant la précision.

10.4.2.2 *CONSTRUCTION DE LA MATRICE PONDEREE*

Afin de considérer les préférences des opérateurs (*cf.* Section 10.4.1.1), les qualités moyennes $QoiS_{moy}$ des configurations exécutables sont alors pondérées.

10.4.2.3 *DEFINITION DES ALTERNATIVES ARTIFICIELLES*

Comme défini dans le Chapitre 8, le prototype TOPSiQC permet de classer, dans un espace géométrique, les configurations du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$ ayant la plus courte distance du contexte qualité de l'opérateur, noté $A^*_{i=1..5}$, et la plus grande distance de la négation du contexte qualité de l'opérateur, noté $A'_{i=1..5}$. Le Tableau 31 résume les valeurs de $A^*_{i=1..5}$ et de $A'_{i=1..5}$ correspondant aux différents contextes qualité définis plus haut (*cf.* Section 10.4.1.1).

Par exemple, pour le contexte qualité *Cq3*, l'opérateur souhaite satisfaire la précision et la confidentialité avec le seuil de satisfaction « + » (normalisé par la valeur « *1* »). Il privilégie la précision (*i.e.* la priorité = 0.7) à la confidentialité (*i.e.* la priorité = 0.3). Ainsi, A_3*= {*0.7, 0.3*} et le contraire du contexte qualité A_3'= {*-0.7, -0.3*}.

Tableau 31. Les valeurs de $A^*_{i=1..5}$ et $A'_{i=1..5}$

$Cq_{i=1..5}$	$A^*_{i=1..5}$	$A'_{i=1..5}$
Cq1={<Précision,+,1>}	A_1*= {*1*}	A_1'= {*-1*}
Cq2={<Confidentialité,+,1>}	A_2*= {*1*}	A_2'= {*-1*}
Cq3={<Confidentialité,+,0.3>, <Précision,+,0.7>}	A_3*= {*0.7, 0.3*}	A_3'= {*-0.7,- 0.3*}
Cq4={< Précision,+,0.3>, <Temps,+,0.7>}	A_4*= {*0.7, 0.3*}	A_4'= {*-0.7,- 0.3*}
Cq5={< Précision,+,0.3>, <Temps,+,0.4>, <Confidentialité ,+,0.3>}	A_5*= {*0.3, 0.4, 0.3*}	A_5'= {*-0.3, -0.4, -0.3*}

10.4.2.4 *CALCUL DES DISTANCES EUCLIDIENNES*

Le prototype TOPSiQC calcule les distances euclidiennes $S^*_{i/i=1..5}$ et $S'_{i/i=1..5}$ de chaque configuration du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$ (*cf.* Chapitre 9). La distance $S^*_{i/i=1..5}$ permet de mesurer, dans un espace géométrique, la distance des configurations par rapport à $A^*_{i/i=1..5}$, tandis que la distance $S'_{i/i=1..5}$ permet de calculer la distance des configurations par rapport à $A'_{i/i=1..5}$.

10.4.2.5 *CALCUL DE LA PROXIMITE (OU LA SIMILARITE) RELATIVE.*

Ces deux distances euclidiennes (*i.e.* $S^*_{i/i=1..5}$ et $S'_{i/i=1..5}$) permettent ensuite de calculer la proximité relative $C^*_{i/i=1..5}$ de chaque configuration de $\mathbf{S}_{\text{Réaliser analyse plasma}}$ par rapport à $A^*_{i/i=1..5}$ et à $A'_{i/i=1..5}$. La proximité relative $C^*_{i/i=1..5}$ correspond au degré de préférence des configurations par rapport aux différents contextes qualité des opérateurs (*cf.* Section 10.4.1.1). Enfin, le prototype TOPSiQC classe dans un ordre descendant les configurations exécutables du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, en fonction du degré de préférence $C^*_{i/i=1..5}$.

10.4.3 ETAPE 3 : DEFINITION DES RESULTATS DE TOPSIQC

Le prototype TOPSiQC permet de classer les configurations du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, en fonction de ce que souhaite l'opérateur, en termes de qualité. Nous présenterons dans ce qui :

1. Les résultats obtenus
2. L'interprétation de ces résultats.

10.4.3.1 *RESULTATS*

Deux résultats sont mis en évidence par le prototype TOPSiQC : le premier concerne le classement des configurations, et le dernier correspond aux configurations non-exécutables (celles-ci ne sont pas classées). Les Figure 147 et Figure 148 illustrent, respectivement, le classement des configurations (une configuration correspond à *Path* dans les Figure 147 et Figure 148) exécutables dans le cas de *Cq5*, ainsi que celles qui ne le sont pas.

La Figure 147 présente le classement des configurations exécutables, fournis par le prototype TOPSiQC. Ce dernier affiche les distances $S^*_{i/i=1..5}$ et $S'_{i/i=1..5}$, ainsi que les proximités relatives $C^*_{i/i=1..5}$ des configurations du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$. Les distances $S^*_{i/i=1..5}$ et $S'_{i/i=1..5}$ précisent l'écart que possède chaque configuration par rapport à, respectivement, $A^*_{i/i=1..5}$ et $A'_{i/i=1..5}$, tandis que les proximités relatives $C^*_{i/i=1..5}$ fixent le degré de préférence de chaque configuration, par rapport au contexte qualité de l'opérateur (*i.e.* $A^*_{i/i=1..5}$). La Figure 147 met en évidence le classement, dans un ordre descendant, des configurations du service $\mathbf{S}_{\text{Réaliser analyse plasma}}$, suivant le degré de préférence $C^*_{i/i=1..5}$: les configurations les mieux classées sont celles qui répondent à deux conditions : (1) elles possèdent la plus petite distance du contexte qualité de l'opérateur (*i.e.* $S^*_{i/i=1..5}$) ; et (2) elles ont la plus grande distance de la négation de ce contexte qualité (*i.e.* $S'_{i/i=1..5}$).

La Figure 148, quant à elle, met en évidence les configurations non exécutables (correspond à la colonne *invalidates* à la Figure 148), retournées par le prototype TOPSiQC. Aussi, ce dernier affiche les contraintes de dépendance qui ne sont pas satisfaites par chacune des configurations non exécutables. Celles-ci ne sont pas prises en compte lors de la comparaison des configurations.

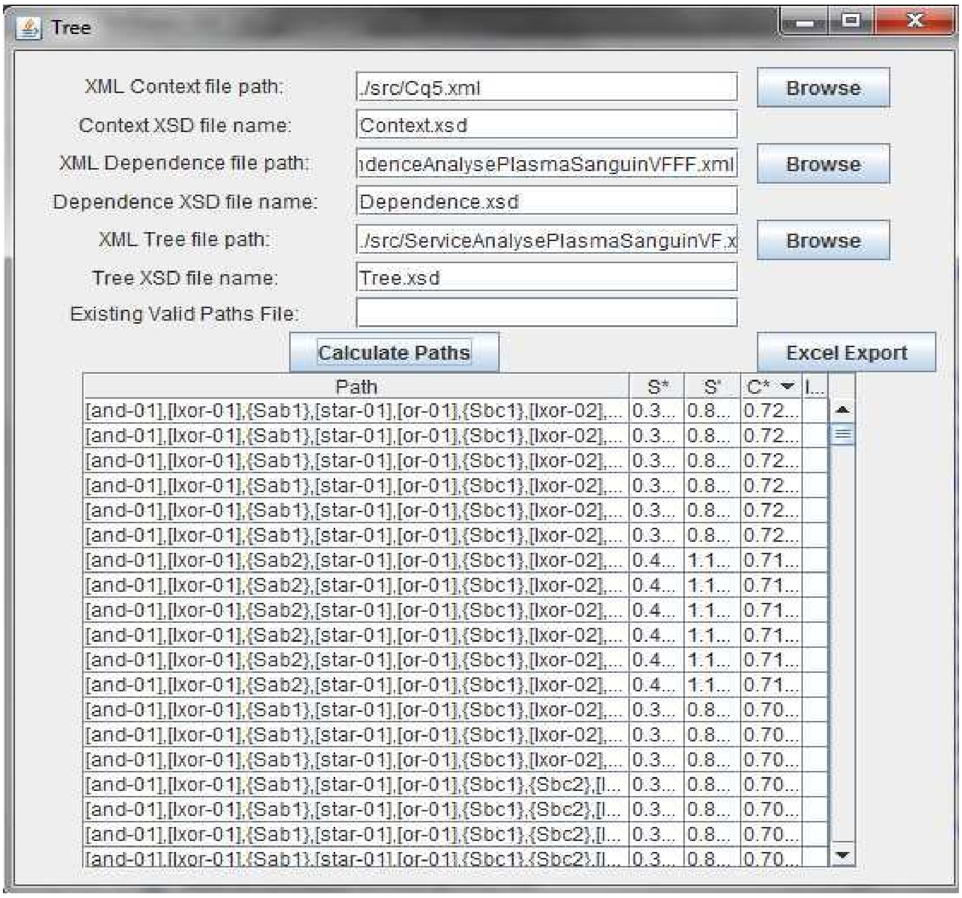

Figure 147. Les configurations exécutables de $\mathbf{S}_{\text{Réaliser analyse plasma}}$ dans le cas de *Cq5*

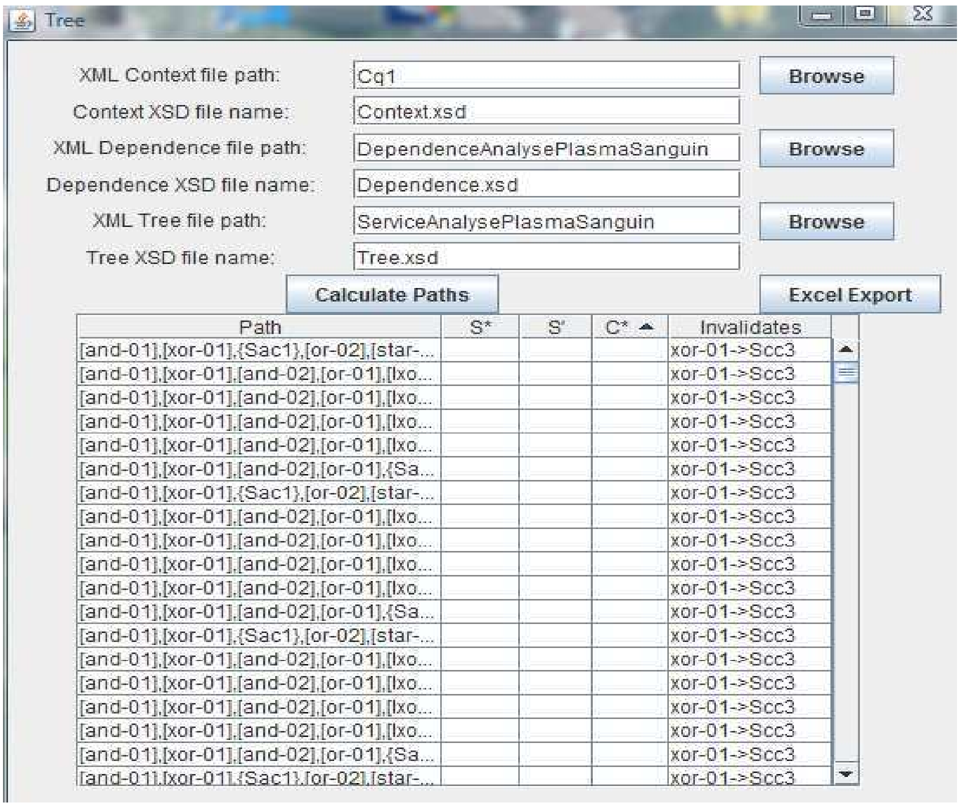

Figure 148. Les configurations non-exécutables de $\mathbf{S}_{\text{Réaliser analyse plasma}}$

10.4.3.2 *INTERPRETATION DES RESULTATS*

Dans la perspective de réaliser une analyse de plasma sanguin (*i.e.* le but *Réaliser analyse plasma*), la recherche du meilleur compromis entre les qualités que l'opérateur souhaite satisfaire et celles offertes par les services, est déterminée par les proximités relatives $C^*_{i/i=1..5}$, calculées par le prototype TOPSiQC. Les proximités relatives correspondent au degré de préférence que possède chaque configuration exécutable du service agrégat $\mathbf{S}_{\text{Réaliser analyse plasma}}$ par rapport aux différents contextes qualité de l'opérateur (*i.e.* $Cq_{i/i=1..5}$).

Néanmoins, le calcul des proximités relatives se fait uniquement sur des configurations exécutables. En effet, le prototype TOPSiQC vérifie l'exécutabilité des configurations générées, en considérant les contraintes de dépendance définies sur le service $\mathbf{S}_{\text{Réaliser analyse plasma}}$ (*cf.* Section 9.3.1.1). Par exemple, la configuration $\mathbf{Conf}_1=\bullet(\mathbf{S}_{ab1},\mathbf{S}_{bb1},\mathbf{S}_{bc1},\mathbf{S}_{bc2},\mathbf{S}_{cd1},\mathbf{S}_{de6},\mathbf{S}_{ef1},\mathbf{S}_{fg2},\mathbf{S}_{gh1})$ précise que l'automate implémente l'authentification par mot de passe, et effectue un chargement des tubes par mode urgent (noté $\mathbf{S}_{bb1}$). L'automate peut effectuer également une identification automatique (noté $\mathbf{S}_{bc1}$) et par lecture de code à barre (noté $\mathbf{S}_{bc2}$) des tubes. Aussi, cet automate réalise le test TS (noté $\mathbf{S}_{de6}$), il effectue une analyse sans incubation (noté $\mathbf{S}_{ef1}$), et il utilise une méthode immunologique (noté $\mathbf{S}_{fg2}$). Néanmoins, la configuration $\mathbf{Conf}_1$ n'est pas exécutable, car elle ne satisfait pas les contraintes de dépendance suivantes : $\boldsymbol{S}_{bb1} \Rightarrow \boldsymbol{S}_{bc3}$ et $\boldsymbol{S}_{ef1} \Rightarrow \neg$ lxor-03, malgré qu'elle satisfait les contraintes $\boldsymbol{S}_{de6} \Rightarrow \neg \boldsymbol{S}_{ef2}$ et $\boldsymbol{S}_{de6} \Rightarrow \neg \boldsymbol{S}_{ef3}$. En effet, lorsque le chargement des tubes est effectué par mode urgent, leur identification doit se faire manuellement (*i.e.* $\boldsymbol{S}_{bb1} \Rightarrow \boldsymbol{S}_{bc3}$). Or, la configuration $\mathbf{Conf}_1$ fait une identification automatique et par lecture de code à barre des tubes, alors qu'elle propose un chargement urgent. Egalement, l'analyse de plasma sans incubation ne peut se faire qu'avec une méthode chronométrique, excluant ainsi les deux autres méthodes de mesure, *i.e.* l'immunologie et la colorimétrie (*i.e.* $\boldsymbol{S}_{ef1} \Rightarrow \neg$ lxor-03).

Le prototype TOPSiQC permet ainsi de détecter toutes les configurations ne pouvant être exécutées. Le Tableau 32 met en évidence quelques configurations non exécutables, ainsi que les contraintes de dépendance non satisfaites par ces configurations. Par exemple, dans la configuration $\mathbf{Conf}_2$, on ne voit pas la présence du service $\mathbf{S}_{fg3}$, alors que le choix des services $\mathbf{S}_{de1}$ et $\mathbf{S}_{de4}$ l'impose, par les contraintes $\mathbf{S}_{de1} \Rightarrow \mathbf{S}_{fg3}$ et $\mathbf{S}_{de4} \Rightarrow \mathbf{S}_{fg3}$. Aussi, la présence des services $\mathbf{S}_{de6}$, $\mathbf{S}_{de7}$ et $\mathbf{S}_{ef1}$ excluent la présence des services $\mathbf{S}_{ef2}$, $\mathbf{S}_{ef3}$ et lxor-03, dans la même configuration.

Tableau 32. Quelques configurations non exécutables

Configurations non exécutables	***Contrainte de dépendance***
Conf$_1$=●(Sab1,Sbb1,Sbc2,Scd2,Sde1,Sef3,Sfg3,Sgh1)	Sbb1 $\Rightarrow$ Sbc3
Conf$_2$=●(Sab1,Sbc1,Sbc2,Sbc3,Scd1,Sde1,Sde2,Sde3,Sde4,Sde5,Sde6,Sde7,Sef1,Sef2,Sef3,Sfg1,Sfg2,Sgh2)	Sde4 $\Rightarrow$ Sfg3 Sde1 $\Rightarrow$ Sfg3 Sde6 $\Rightarrow \neg$ Sef2 Sde6 $\Rightarrow \neg$ Sef3 Sde7 $\Rightarrow \neg$ Sef2 Sde7 $\Rightarrow \neg$ Sef3 Sef1 $\Rightarrow \neg$ lxor-03
Conf$_3$=●(Sab1,Sbb1,Sbc1,Sbc2,Scd2,Sde1,Sde2,Sde3,Sde4,Sde5,Sde6,Sde7,Sef1,Sef2,Sef3,Sfg1,Sfg2,Sgh1)	Sbb1 $\Rightarrow$ Sbc3 Sde4 $\Rightarrow$ Sfg3 Sde1 $\Rightarrow$ Sfg3 Sde6 $\Rightarrow \neg$ Sef2 Sde6 $\Rightarrow \neg$ Sef3 Sde7 $\Rightarrow \neg$ Sef2 Sde7 $\Rightarrow \neg$ Sef3 Sef1 $\Rightarrow \neg$ lxor-03

Une fois les configurations non exécutables identifiées, le prototype TOPSiQC permet par la suite de comparer les différentes configurations exécutables retenues (*i.e.* celles qui satisfont les contraintes de dépendance). Le prototype TOPSiQC classe ces configurations selon leurs proximités relatives (*i.e.* $C^*_{i/i=1..5}$). Nous discutons dans ce qui suit les différents résultats obtenus, compte tenu des différents contextes qualité $Cq_{i/i=1..5}$.

Le contexte qualité Cq1

Dans le cas de *Cq1*, l'opérateur souhaite avoir, lors de l'analyse de plasma sanguin, les configurations lui permettant de lui fournir de la précision (*cf.* Figure 139).

Les résultats obtenus montrent que les configurations (*cf.*Figure 149) qui répondent complètement au contexte *Cq1* (*i.e.* $C^*_1=1$) sont celles qui effectuent une identification automatique (*i.e.* le service $\mathbf{S}_{bc1}$ appartient aux configurations) et/ou par lecture de code à barre (*i.e.* le service $\mathbf{S}_{bc2}$ appartient aux configurations) des tubes, alors que l'identification manuelle (*i.e.* le service $\mathbf{S}_{bc3}$) contribue négativement à la précision (*cf.* Annexe 2). Les configuration qui répondent complètement au contexte *Cq1* font de la récupération du plasma par microfiltration (*i.e.* le service $\mathbf{S}_{cd2}$ appartient aux configurations), et enfin elles terminent le processus d'analyse par une validation des résultats (*i.e.* le service $\mathbf{S}_{gh1}$ appartient aux configurations).

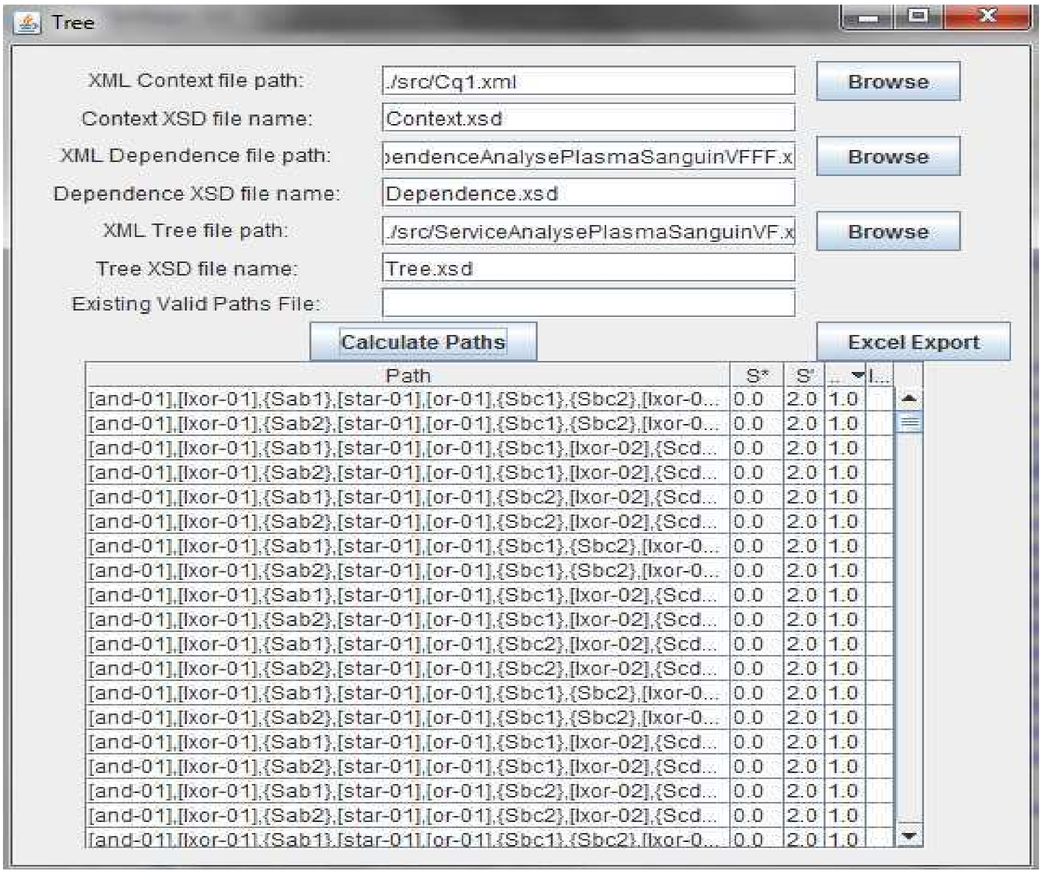

Figure 149. Les meilleures configurations de $\mathbf{S}_{\text{Réaliser analyse plasma}}$, dans le cas de *Cq1*

Le contexte qualité Cq2

Dans le cas de *Cq2*, l'opérateur souhaite réaliser une analyse de plasma sanguin en satisfaisant (*i.e.* seuil de satisfaction est « + ») la confidentialité (*cf.* Figure 140).

Les résultats obtenus montrent que les configurations (*cf.*Figure 150) qui répondent complètement au contexte *Cq2* (*i.e.* $C^*_2=1$) sont celles qui effectuent une authentification par mot de passe de l'opérateur (*i.e.* le service $\mathbf{S}_{ab1}$ appartient aux configurations).

Nous notons que l'authentification de l'opérateur par clé (*i.e.* le service $\mathbf{S}_{ab2}$) fournit une meilleure confidentialité (*i.e.* « ++ »), mais les configurations comprenant ce service sont classées en deuxième position (*i.e.* C^*_2=0.75), derrière celles qui contiennent le service d'authentification par mot de passe. Autrement dit, le prototype TOPSiQC classe en première position les configurations qui répondent au mieux au contexte qualité (*i.e.* la confidentialité avec le seuil « + »), et non celles qui possèdent la meilleure qualité (*i.e.* la confidentialité avec le seuil « ++ »). En d'autres termes, les préférences de l'opérateur priment sur les caractéristiques du service.

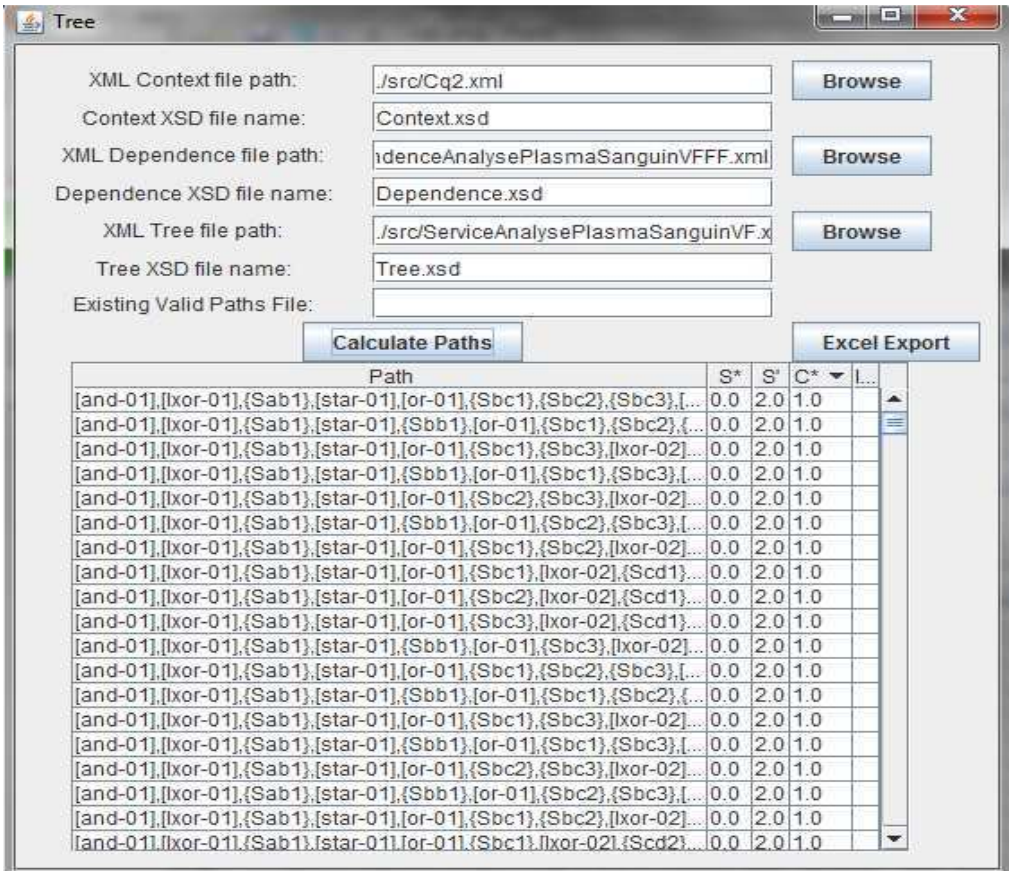

Figure 150. Les meilleures configurations de $\mathbf{S}_{\text{Réaliser analyse plasma}}$, dans le cas de *Cq2*

Le contexte qualité Cq3

Dans le cas de *Cq3*, l'opérateur souhaite réaliser une analyse de plasma sanguin en satisfaisant (*i.e.* seuil de satisfaction est « + ») la confidentialité et la précision. L'opérateur privilégie la Précision, avec une priorité de *0,7*, à la Confidentialité, avec une priorité de *0,3* (*cf.* Figure 141).

Les résultats obtenus montrent que certaines configurations (*cf.* Figure 151) répondent complètement au contexte *Cq3* (*i.e.* $C^*_3=1$) :

- elles sont composées du service effectuant l'authentification par mot de passe de l'opérateur (*i.e.* le service $\mathbf{S}_{ab1}$ appartient aux configurations) ;
- elles effectuent une identification automatique (*i.e.* le service $\mathbf{S}_{bc1}$ appartient aux configurations) et/ou par lecture de code à barre (*i.e.* le service $\mathbf{S}_{bc2}$ appartient aux configurations) des tubes ;
- elles font la récupération du plasma par microfiltration (*i.e.* le service $\mathbf{S}_{cd2}$ appartient aux configurations) ; et enfin
- elles terminent le processus d'analyse par une validation des résultats (*i.e.* le service $\mathbf{S}_{gh1}$ appartient aux configurations).

Dans le cas de *Cq3*, la précision et la confidentialité ne sont pas des qualités conflictuelles : le service fournissant la confidentialité n'entrave pas la précision, et les services offrant la précision ne gênent pas la confidentialité.

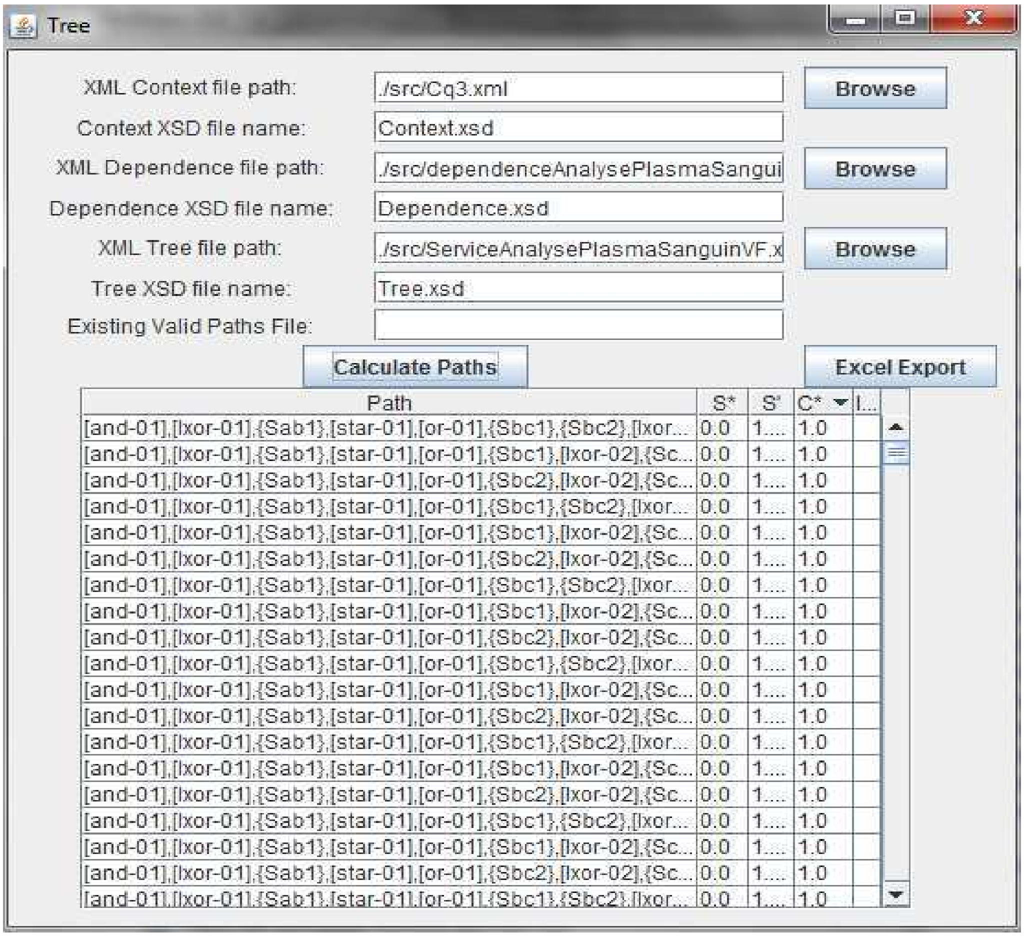

Figure 151. Les meilleures configurations de $\mathbf{S}_{\text{Réaliser analyse plasma}}$, dans le cas de *Cq3*

Le contexte qualité Cq4

Dans le cas de *Cq4*, l'opérateur désire réaliser une analyse de plasma sanguin en satisfaisant (*i.e.* seuil de satisfaction est « + ») simultanément le Temps, avec une priorité de *0,7* et la Précision, avec une priorité de *0,3* (*cf.* Figure 142).

Les résultats obtenus montrent qu'il n'existe pas de configurations qui répondent complètement au contexte *Cq4* (*i.e.* C^*_4=0,718), car les deux qualités (*i.e.* le temps et la précision) sont conflictuelles. Vu que l'opérateur privilégie la performance (*i.e.* le temps) à la précision, les configurations (*cf.* Figure 152) les mieux classées sont alors celles qui privilégient la performance, à savoir :

- elles réalisent une authentification par mot de passe ou par clé. Les deux authentifications influencent négativement (« - ») le Temps, mais restent obligatoires (grâce à l'opérateur de composition « ● ») dans le processus d'analyse.

- elles effectuent une identification automatique des tubes (*i.e.* le service $\mathbf{S}_{bc1}$ appartient aux configurations). Ce type d'identification est rapide et précis. L'identification par lecture de code à barre est également précise, mais moins rapide. Cette identification ne fait pas partie des meilleures configurations, car l'opérateur privilégie la rapidité à la précision.

- elles font la récupération du plasma par centrifugation (*i.e.* le service $\mathbf{S}_{cd1}$ appartient aux configurations). Celle-ci est plus rapide que la récupération par microfiltration, mais moins précise. Aussi, la microfiltration est exclue des meilleures configurations, car l'opérateur préfère la rapidité à la précision.

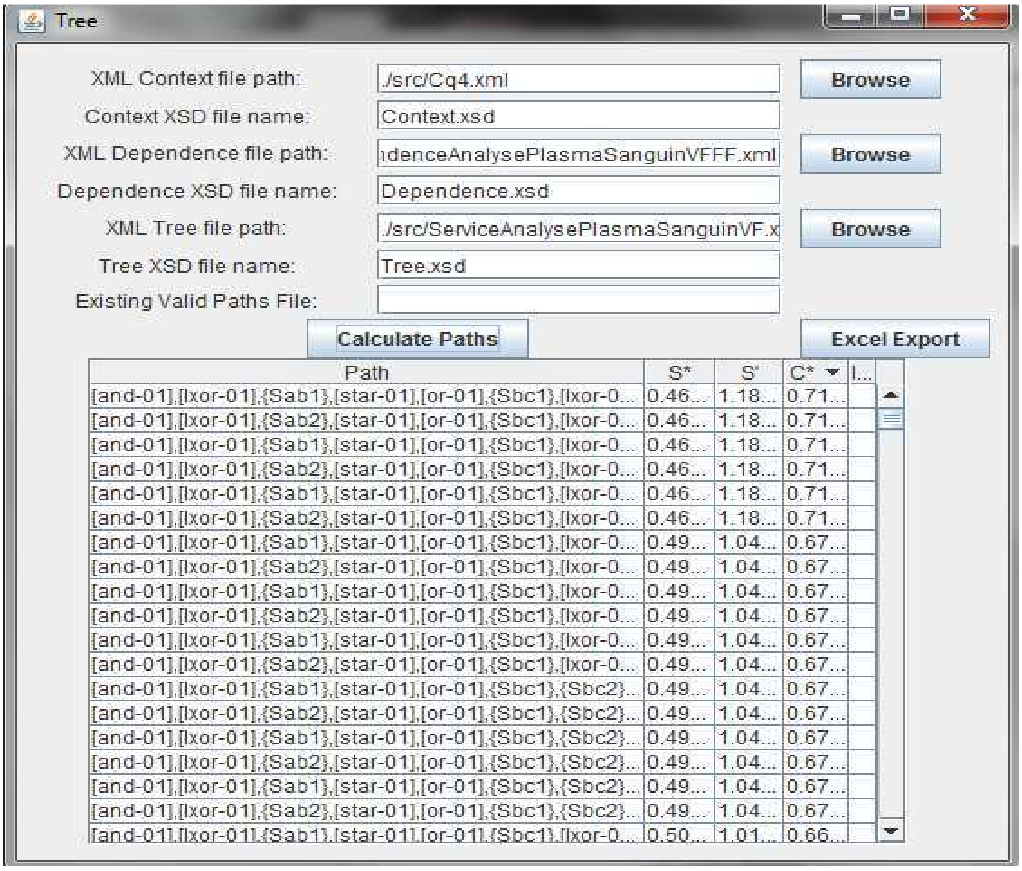

Figure 152. Les meilleures configurations de $\mathbf{S}_{\text{Réaliser analyse plasma}}$, dans le cas de *Cq4*

- elles effectuent des tests sans incubation (*i.e.* le service $\mathbf{S}_{ef1}$ appartient aux configurations), et utilisent la méthode de mesure chronométrique (*i.e.* le service $\mathbf{S}_{fg1}$ appartient aux configurations). Ces services contribuent positivement (« + ») au Temps.

- et elles terminent le processus d'analyse par complétude (*i.e.* le service $\mathbf{S}_{gh2}$ appartient aux configurations). Ces configurations sont plus rapides que celles qui terminent le processus par validation, mais moins précise.

Dans le cas de *Cq4*, la précision et la performance, via le temps, sont des qualités conflictuelles : les services fournissant la précision diminuent la performance. Afin d'effectuer un compromis entre ces qualités, le prototype exploite les priorités octroyées par

l'opérateur à chacune des qualités. L'opérateur privilégie la performance à la précision, et les meilleures configurations obtenues sont composées alors des services qui privilégient la performance à la précision. Par exemple, l'identification par lecture de code à barre est précise, mais moins performante. Cette identification ne fait pas partie des meilleures configurations, car l'opérateur privilégie la performance à la précision.

Le contexte qualité Cq5

Dans le cas de *Cq5*, l'opérateur désire réaliser une analyse de plasma sanguin en satisfaisant (*i.e.* seuil de satisfaction est « + ») simultanément toutes les qualités, à savoir le Temps, avec une priorité de *0,4*, la Précision et la Confidentialité avec une priorité de *0,3* (*cf.* Figure 143).

Pareil que pour le *Cq4*, les résultats obtenus montrent qu'il n'existe pas de configurations qui répondent complètement au contexte *Cq5* (*i.e.* C^*_5=0,729), car les trois qualités (*i.e.* le temps, la confidentialité et la précision) ne peuvent pas être satisfaites simultanément. Etant donné que l'opérateur privilégie la rapidité (*i.e.* le temps) à la confidentialité et à la précision, les configurations les mieux classées ; elles ne sont que six dans ce cas (*cf.* Figure 147), sont donc celles qui offrent un compromis entre ces trois qualités, tout en considérant les priorités octroyées à chacune d'elles. Les meilleures configurations sont alors celles :

- qui réalisent une authentification par mot de passe de l'opérateur. Les deux authentifications (par mot de passe et par clé) influencent négativement (« - ») le Temps, mais l'authentification par mot de passe répond au besoin de sécurité de l'opérateur (*i.e. Cq5*), car elle contribue positivement (« + ») à la confidentialité.

- qui effectuent une identification automatique des tubes (*i.e.* le service $\mathbf{S}_{bc1}$ appartient aux configurations). Celle-ci contribue positivement (« + ») au Temps et à la Précision.

- qui font la récupération du plasma par centrifugation (*i.e.* le service $\mathbf{S}_{cd1}$ appartient aux configurations) ou par microfiltration (*i.e.* le service $\mathbf{S}_{cd2}$ appartient aux configurations). Les configurations qui font de la centrifugation sont plus rapides que celles qui font de la microfiltration. Cependant, la récupération de plasma par microfiltration contribue positivement (« + ») à la Précision, contrairement à la centrifugation.

- qui effectuent des tests sans incubation (*i.e.* le service $\mathbf{S}_{ef1}$ appartient aux configurations), et utilisent la méthode de mesure chronométrique (*i.e.* le service $\mathbf{S}_{fg1}$ appartient aux configurations). Ces services contribuent positivement (« + ») au Temps.

- et elles terminent le processus d'analyse par complétude (*i.e.* le service $\mathbf{S}_{gh2}$ appartient aux configurations) ou par validation des résultats (*i.e.* le service $\mathbf{S}_{gh1}$ appartient aux configurations). Terminer le processus par complétude contribue positivement (« + ») au

Temps, contrairement à l'arrêt du processus par validation. Néanmoins, ce dernier est plus précis.

Dans le cas de *Cq5*, l'opérateur souhaite satisfaire la confidentialité, la précision et la performance, via le temps. La performance est en conflit avec la confidentialité et la précision : les services fournissant la précision (*e.g.* $\mathbf{S}_{gh1}$) ou offrant de la confidentialité (*e.g.* $\mathbf{S}_{ab1}$) diminuent la performance. Afin d'effectuer un compromis entre ces qualités, le prototype tient compte des priorités octroyées par l'opérateur à chacune des qualités. L'opérateur privilégie la performance, avec une priorité de *0,4*, à la précision et à la confidentialité qui sont au même pied d'égalité, avec une priorité de *0,3*. Les meilleures configurations obtenues sont composées alors des services qui privilégient la performance, tels que les services $\mathbf{S}_{bc1}$, $\mathbf{S}_{ef1}$ et $\mathbf{S}_{fg1}$. Elles contiennent également le service d'authentification par mot de passe (*i.e.* $\mathbf{S}_{ab1}$) qui contribue négativement au Temps, mais qui reste un service obligatoire dans le processus. Enfin, pour fournir le compromis entre les qualités, les meilleures configurations sont celles qui font de la récupération de plasma par centrifugation (*i.e.* $\mathbf{S}_{cd1}$ contribue « + » au Temps et « - » à la Précision) et terminent le processus par la validation (*i.e.* $\mathbf{S}_{gh1}$ contribue « - » au Temps et « + » à la Précision), ou celles qui récupèrent le plasma par microfiltration (*i.e.* $\mathbf{S}_{cd2}$ contribue « - » au Temps et « + » à la Précision) et arrêtent le processus par complétude (*i.e.* $\mathbf{S}_{gh2}$ contribue « + » au Temps et « - » à la Précision). Nous pouvons dire que la présence des deux services (($\mathbf{S}_{cd1}$ et $\mathbf{S}_{gh1}$) ou ($\mathbf{S}_{cd2}$ et $\mathbf{S}_{gh2}$)) ensemble dans les meilleures configurations permet d'avoir des qualités neutres concernant le Temps et la Précision.

Nous remarquons que les résultats du prototype sont en adéquation avec les préférences de l'opérateur. En effet, les résultats founis pour chaque *Cq* reflètent le degré de préférence de l'opérateur, qui est fortement lié aux priorités octroyées aux qualités. Autrement dit, les résultats sont différents dans *Cq4* et *Cq5*, malgré que le Temps est privilégié à la Précision dans les deux contextes. Néanmoins, c'est la différence des priorités (*i.e.* priorité de *0,7* au Temps et de *0,3* à la précision dans le cas de *Cq4*, et priorité de *0,4* au Temps et de *0,3* à la précision dans le cas de *Cq5*) qui a impliqué cette différence de résultat.

Nous pouvons dire que la configuration intentionnelle que nous proposons est orientée vers l'utilisateur final. En effet, on a montré, par l'exemple de l'analyse de plasma sanguin, que les préférences de l'opérateur, *i.e.* ses différents contextes qualité priment dans les choix des variantes qui composent les différentes configurations de l'automate. Autrement dit, selon les

différents contextes qualité, le prototype TOPSiQC a proposé différentes configurations à l'opérateur de l'automate.

10.5 *CONCLUSION*

Ce chapitre a présenté l'application de la méthodologie de sélection et de configuration intentionnelle des services à l'exemple du processus d'analyse de sang. Particulièrement, dans le cas d'un grand laboratoire d'analyse, l'opérateur doit sélectionner, parmi les configurations d'un automate du laboratoire, celles qui doivent répondre au mieux à ses exigences non-fonctionnelles, en termes de temps, de précision et de confidentialité.

L'application de la méthode de modélisation de la qualité de service a permis de :

- montrer la faisabilité d'une modélisation explicite de la qualité du service intentionnel par l'intégration des buts qualité à la carte *Réaliser analyse plasma*. La modélisation de la QoiS a pu être effectuée.
- montrer la faisabilité du calcul des différentes QoiS : la QoiS simple et la $QoiS_{moy}$.
- montrer la faisabilité d'un référentiel qualité dans le domaine du diagnostic *in vitro* de l'hémostase, afin de partager les connaissances concernant les qualités.
- montrer la faisabilité de l'association des contraintes de dépendance aux services afin de gérer la variabilité des services, ainsi que la génération de configurations exécutables de services.

L'application de la méthode de configuration intentionnelle des services a permis de :

- montrer que la formalisation des exigences des utilisateurs dans un contexte qualité permet de faciliter la configuration.
- montrer la possibilité de choisir parmi les configurations d'un service agrégat, celles qui se rapprochent le plus du contexte qualité.
- montrer la possibilité, pour un même besoin fonctionnel, de varier les besoins de qualité et retrouver à chaque fois les configurations les plus adéquates.
- montrer la faisabilité de notre solution grâce à l'utilisation de TOPSiQC.

L'application de la solution que nous proposons au processus d'analyse de plasma sanguin dans un laboratoire d'analyse a mis en évidence l'utilité de permettre aux opérateurs du laboratoire de choisir les automates permettant de répondre au mieux aux besoins des opérateurs.

Pour réaliser cet exemple, nous avons utilisé un serveur doté des caractéristiques suivantes : processeur quadri-cœur *I7* d'Intel, *64* bits, avec Windows *7* en *64* bits, *6* GO de RAM et un disque dur de 10000tr/min.

Les traitements de TOPSiQC (génération de configurations, vérification des contraintes et comparaison des configurations) ont pris environ *3* min pour chaque contexte qualité traité (cinq contextes au total). Néanmoins, on n’est pas loin des limites du prototype, car on est à *4* GO de RAM sur *6* GO : TOPSiQC prend *2* GO pour le calcul des configurations et *2* GO pour le calcul des configurations exécutables/non-exécutables. Le calcul des configurations est exponentiel à cause de l’existence des relations de type *Ou,* qui multiplie le nombre de configurations de façon exponentiel et limite ainsi le prototype.

Pour améliorer la performance du prototype TOPSiQC, on pourrait sauvegarder les configurations exécutables dans un fichier à part, et effectuer, à partir de ce fichier, la comparaison des configurations pour chaque contexte qualité. Cette façon de faire va améliorer sensiblement la performance du prototype. Aussi, on pourrait utiliser la programmation par contraintes.

CHAPITRE 11 : CONCLUSION

11.1 CONTRIBUTIONS

Nous avons abordé dans cette thèse deux perspectives dans la sélection des services : une perspective purement technique, promue par la communauté SOC, laquelle s'intéresse à l'exécution des services logiciels. Néanmoins, le processus de sélection lui-même, la description des services et de leurs qualités restent incompréhensibles aux agents métier. D'autre part, une perspective de plus haut niveau, promue par les approches de la communauté RE, laquelle s'intéresse aux agents métier et à leurs exigences fonctionnelles et non-fonctionnelles. Ces approches définissent des services et des qualités de haut niveau, et une sélection de ces services lors de la conception des systèmes. Néanmoins, elles restent déconnectées des services logiciels et de leur exécution. Dans cette thèse, nous avons proposé de relier ces deux perspectives afin d'effectuer une sélection des services de haut niveau en fonction des exigences non-fonctionnelles des agents métier.

Afin de mettre en correspondance les exigences fonctionnelles des agents métier avec les descriptions des services logiciels, nous avons adopté l'architecture orienté service intentionnel (*iSOA- intentional Service Oriented Architecture*). L'architecture iSOA permet de déplacer la vision orientée fonction du service vers une vision orientée intention. Le service intentionnel de l'architecture iSOA est un service de haut niveau, dédié à l'utilisateur. Il permet de décrire le service en mettant en évidence le but métier que le service permet de réaliser, plutôt que la description technique. Ainsi, les interfaces de ces services sont décrites dans un mode intentionnel, publiées dans un annuaire de services intentionnels, et peuvent être découvertes par des agents métier. Aussi, le service intentionnel est opérationnalisé par des services logiciels. Néanmoins, dans iSOA, le service intentionnel correspond aux exigences fonctionnelles des agents métier, sans autant considérer ses exigences non-fonctionnelles, lesquelles jouent un rôle prépondérant dans la sélection des services. Nous avons proposé alors dans cette thèse de doctorat une extension de l'architecture iSOA en mettant en œuvre une méthode de sélection intentionnelle des services, en fonction des exigences non-fonctionnelles des agents métier.

La méthode que nous avons proposée dans cette thèse se base sur le méta-modèle du référentiel qualité. Le référentiel qualité permet de représenter conjointement les exigences

non-fonctionnelles du niveau stratégique et du niveau opérationnel. Le référentiel qualité s'appuie sur le cadre de référence NFR (*NFR-framework*) (Chung et *al.*,2000) pour définir formellement les exigences non-fonctionnelles comme des buts qualité à partiellement satisfaire. Le référentiel qualité fournit, par domaine particulier, les buts qualité (*i.e. soft goal*) envisageables, les buts (*i.e. hard goal*) contribuant à satisfaire partiellement ces buts qualité, ainsi que les métriques permettant de quantifier les buts qualité. A travers ces métriques, le référentiel qualité permet d'associer à des seuils de satisfaction prédéfinis une évaluation quantitative, moins subjective et qui peut être directement reliée aux services logiciels. Le référentiel qualité permet aux fournisseurs et aux agents métier de partager les mêmes concepts et la même sémantique concernant les seuils de satisfaction : les fournisseurs pour décrire la qualité de leurs services et les agents métier pour formaliser leurs exigences non-fonctionnelles dans un contexte qualité. Le référentiel qualité promeut un consensus qualité dans un domaine particulier et améliore la crédibilité des fournisseurs.

La deuxième contribution de cette thèse est le modèle *MiS-q* lequel définit la qualité du service intentionnel (*QoiS-Quality of intentional Service*) comme l'ensemble des buts qualité que le service intentionnel permet partiellement de satisfaire. La QoiS simple est associée au service atomique, tandis que la QoiS globale correspond au service agrégat. Comme la QoiS globale n'est pas directement exploitable par le processus de sélection, nous avons proposé alors de calculer la $QoiS_{min}$ pour comparer les services agrégats, et la $QoiS_{moy}$ pour comparer les configurations du service agrégat.

Le service intentionnel introduit de la variabilité dans la manière d'atteindre le but associé à un service et, donc dans la manière de satisfaire partiellement les buts qualité. Telle qu'une famille de produits logiciels, la variabilité du service intentionnel, permet de mettre en évidence les caractéristiques communes à tout processus réalisant un même but, tout en rendant explicite les variations dans ce processus. La variabilité encapsulée dans le service intentionnel nous a incités à définir les contraintes de dépendance pouvant exister entre les variantes d'un service agrégat. Les contraintes permettent de valider l'exécutabilité d'un service agrégat donné. Elles s'appliquent aussi bien à des services atomiques, qu'agrégats, dans le cas d'un service à variation. Le modèle *MiS-q* permet de représenter les contraintes *Exclut* et *Exige* : la contrainte Exclut définit que l'exécution d'un service écarte l'exécution d'un autre service, tandis que la contrainte Exige définit que l'exécution d'un service requiert l'exécution d'un autre service. Les variations du service intentionnel concernent non seulement les différentes manières de réaliser le but du service, mais aussi les manières de satisfaire partiellement les buts qualité. Ces variations permettent de répondre aux exigences

fonctionnelles et aux exigences non-fonctionnelles des clients de manière différenciée. Le modèle *MiS-q* permet ainsi d'établir la mise en correspondance au niveau intentionnel entre les exigences non-fonctionnelles des agents métier et la qualité des services logiciels.

Nous avons également défini, dans cette thèse, une méthodologie permettant de construire le modèle *MiS-q* à partir de la carte des besoins et du référentiel qualité. En effet, un service intentionnel est exprimé par une carte des besoins. Dans le cadre de l'architecture iSOA, nous proposons que le fournisseur métier se base sur le référentiel qualité pour associer des qualités aux services métier qu'ils souhaitent publier. Ceci permet d'uniformiser, pour chaque domaine, les buts qualité pouvant être satisfaits partiellement par les services, ainsi que la sémantique des seuils de satisfaction, utilisés conjointement à ces buts qualité. L'intégration des buts qualité à la carte des besoins se fait grâce aux liens de contributions. Chaque section opérationnalisable de la carte peut contribuer à la satisfaction partielle des buts qualité simples à l'aide d'un certain lien de contribution. Un ensemble de règles méthodologiques sont proposées pour générer les différentes QoiS (simple et globale) de la carte des besoins.

La méthode que nous avons proposée dans cette thèse supporte la sélection intentionnelle des services, laquelle permet de guider l'agent métier dans le choix des services répondant le mieux à ses exigences non-fonctionnelles. Afin de diminuer le risque d'incompatibilité lors de la sélection intentionnelle des services, nous avons formalisé les exigences de l'agent métier dans un contexte qualité. Celui-ci regroupe le but fonctionnel (ou *hard goal*) que l'agent métier désire réaliser, ainsi que l'ensemble des buts qualité qu'il souhaite satisfaire. Pour chaque but qualité, le client précise le seuil de satisfaction aux delà duquel il accepte toute satisfaction, ainsi que la priorité associée à cette qualité. Dans le cadre de l'architecture iSOA, l'agent métier aussi se base sur le référentiel qualité pour, d'une part, décider des qualités qu'il souhaite satisfaire et, d'autre part, vérifier la sémantique des seuils de satisfaction, afin qu'il puisse définir ceux qui lui conviennent. La sélection intentionnelle des services consiste à choisir, parmi les différents services qui réalisent le même but, ceux qui correspondent au mieux au contexte qualité des agents métier. Nous avons également proposé de configurer le service agrégat, afin de choisir parmi les configurations exécutables du service agrégat, celles qui sont les plus proches du contexte qualité de l'agent métier. La configuration du service agrégat nous a permis de considérer la variabilité exprimée par ce service.

Nous avons opté pour la méthode d'aide à la décision multicritères TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) pour faire la sélection et la configuration intentionnelle des services. Avec la méthode TOPSIS, les services sélectionnés sont ceux qui possèdent d'une part la plus courte distance du contexte qualité de l'agent métier, et d'autre

part la plus grande distance de la négation du contexte qualité. Nous avons développé le prototype TOPSiQC (*Technique for Order Preference by SImilarity to Quality Context*), lequel correspond à l'implémentation de la méthode TOPSIS, adaptée à la sélection et à la configuration intentionnelle des services. Le développement du prototype TOPSiQC a montré la faisabilité de la solution que nous avons proposée.

11.2 *PERSPECTIVES*

Le travail présenté dans cette thèse peut être poursuivi dans deux directions principales.

Intégrer les différents algorithmes et extension de modèle dans la plateforme iSOA

Nous proposons que les résultats de cette thèse soient intégrés à la plateforme iSOA.

Il s'agit de fusionner l'algorithme de recherche des services par les buts avec l'algorithme de sélection proposé, lequel traite uniquement l'aspect qualité. En effet, la recherche de service consiste à trouver les services réalisant le but métier de l'agent métier, alors que la sélection permet d'affiner la recherche, en trouvant les services satisfaisant partiellement les buts qualité de l'agent métier.

Il s'agit également d'intégrer l'algorithme de configuration dans l'annuaire des services intentionnel. La configuration intentionnelle s'applique sur le service intentionnel qui supporte la variabilité dans la réalisation du but métier. Configurer le service intentionnel consiste à trouver les variantes à l'intérieur du service lui-même qui permettent de satisfaire partiellement les buts qualité de l'agent métier.

Le référentiel qualité doit être aussi intégré à la plateforme iSOA. En effet, le référentiel qualité permet d'établir le consensus qualité entre, d'une part, les fournisseurs métier d'un domaine, et d'autre part les fournisseurs et les agents métier du même domaine.

Appliquer cette démarche à l'ingénierie des méthodes

Nous proposons que la démarche adoptée dans cette thèse soit appliquée à l'ingénierie des méthodes.

Afin de proposer une approche d'ingénierie des méthodes centrée sur l'usage, Iacovelli (Iacovelli,2009) propose le concept de service méthodologique (MaaS), afin de représenter les parties réutilisables d'une méthode. Nous proposons alors d'appliquer la démarche définie dans cette thèse, afin de prendre en compte la qualité des services méthodologiques.

BIBLIOGRAPHIE

(Acher et *al.*, 2010) Acher M., Collet P., Lahire P. and France R. *Managing variability in workflow with feature model composition operators*. In Proceedings of the 9th international conference on Software composition, 2010.

(Aiello&Giorgini,2004) Aiello M. and Giorgini P. *Applying the Tropos Methodology for Analysing Web Services Requirements and Reasoning about Qualities of Services*. Journal for the Informatics Professional, 5(4), 2004.

(AitAliSlimane et *al.*,2009) Ait-Ali-Slimane A., Kirsch Pinheiro M. and Souveyet C. *Goal Reasoning for Quality Elicitation in the ISOA approach*. International Conference on Research Challenges in Information Science (RCIS), IEEE, Maroc, 2009.

(AitAliSlimane et *al.*,2011) Ait-Ali-Slimane A., Kirsch Pinheiro M. and Souveyet C. Considering quality of a service in an intentional approach. To appear in Handbook of Research on Non-Functional Properties for Service-Oriented Systems: Future Directions. 2011.

(Afandi et *al.*,2006) Afandi A., Zhang J. and Gunter C.A. AMPol-Q : Adaptive Middleware Policy to Support QoS. In Service-Oriented Computing (ICSOC), 2006.

(Aljoumaa et *al.*, 2010) Aljoumaa K., Assar S. and Souveyet C. *Publishing intentional services using new annotation for WSDL*. In Proc. of the 12th International Conference on Information Integration and Web-based Applications & Services (iiWAS2010), Paris, France 2010.

(Aljoumaa et *al.*, 2011) Aljoumaa K., Assar S. and Souveyet C. *Reformulating user's queries for Intentional Services Discovery using Ontology-based Approach*. To appear In Proc. of the fourth IFIP International Conference on New Technologies, Mobility and Security (NTMS'2011), Paris, France 2011.

(Alonso et *al.*,2004) Alonso G., Casati F., Kuno H., and Machiraju V. *Web Services: Concepts, Architecture, and Applications*. Springer Verlag (ISBN: 3540440089), 2004.

(Amyot&Mussbacher,2002) Amyot D. and Mussbacher G. *URN : Towards a New Standard for the Visual Description of Requirements*. 3rd SDL and MSC Workshop (SAM02), Aberystwyth, U.K., LNCS 2599, pp.21-37. June 2002.

(Andrews et *al.*,2003) Andrews T. and al. *Business Process Execution Language for Web Services Version 1.1*. (Disponible sur http ://www.ibm.com/developerworks/library/ws-bpel). May 2003.

(Anton et *al.*,1994) Anton A., McCracken W. and Potts C. *Goal Decomposition and Scenario Analysis in Business Process Reengineering*. In Proc. 6th Conference On Advanced Information Systems Engineering (CAiSE'94), Utrecht, Holland, June 1994.

(Anton,1996) Anton A.I. *Goal-based requirements analysis*. In 2nd International Conference on Requirements Engineering (RE'96), pp.136, 1996.

(Aoyama et *al.*, 2003) Aoyama M., Watanabe K., Nishio Yu. and Moriwaki Y. *Embracing requirements variety for e-Governments based on multiple product-lines frameworks.* In Proceeding RE '03 Proceedings of the 11th IEEE International Conference on Requirements Engineering. September 2003.

(Araban&Sterling,2004) Araban S. and Sterling L.S. *Measuring quality of service for contract aware web services*. In First Australian Workshop on Engineering Service-Oriented Systems,pages 54–56, 2004.

(Assar et *al.*, 2000) Assar S., Ben Achour C., Si-Said S. *Un Modèle pour la spécification des processus d'analyse des Systèmes d'Information*. In Proceedings of INFORSID'2000. pp.287-301.2000.

(Azsanjani,2004) Azsanjani A. *Service-oriented modelling and architecture*. (Disponible sur http://www128.ibm.com/developerworks/webservices/library/ws-soa-design1/), November 2004.

(Balédent,2010) Balédent F. *Physiologie de l'hémostase*. Disponible sur le site http://documentation.ledamed.org/IMG/html/doc-10934.html.

(Barros et *al.*,2005) Barros A., Dumas, M. and Oaks P. *Standards for Web Service Choreography and Orchestration : Status and Perspectives*. In Proc. of the 3rd International Conference the Business Process Management (BPM 2005), 1st International Workshop on Web Service Choreography and Orchestration for Business Process Management, Nancy, France, pp.1-15, September 2005.

(Banerjee et *al.*,1987) Banerjee J., Kim W., Kim H-J. and Korth H-F. *Semantics and Implementation of Schema Evolution in Object Oriented Databases*. In Proc. of the ACM-SIGMOD Annual Conference, pages 311--322, San Francisco, CA, May 1987.

(Basili&Musa,1991) Basili V.R. and Musa J.D. *The Future Engineering of Software : A Management Perspective*. Computer, v.24(9), pp.90-96, September 1991.

(Baryannis et *al.*,*2008*) Baryannis G., Carro M., DanylevychO., Dustdar S., Karastoyanova D., Kritikos K., Leitner P., Rosenberg F. and Wetzstein B. *Overview of the State of the Art in Composition and Coordination of Services*. By the S-CUBE consortium, July 2008.

(Batagan et *al.*,2009) Batagan L., Pocovnicu A. and Capisizu S. *E-Service Quality Management*. In Journal of Applied Quantitative Method (JAQM), vol.4, 2009.

(Batory, 2005) Batory D. Feature models, grammars, and propositional formulas. In Software Product Lines Conference, volume 3714 of Lecture Notes in Computer Sciences, pages 7–20. Springer–Verlag, 2005.

(Belton&Gear,1983) Belton V. and Gear T. *On a short-coming of Saaty's method of analytic hierarchies*. Omega, pp.228-230,1983.

(Benatallah et *al.*,2002) Benatallah B., Dumas M., Sheng Q.Z and Ngu A. *Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web Services*. In 18th International Conference on Data Engineering (ICDE'02), 2002.

(Benayoun et *al.*,1971) Benayoun R., Laritchev O., de Mongolfier J., Tegny J. *Linear programming with multiple objective functions : STEP method (STEM).* Math. Program. 1, 3, pp. 366–375. 1971.

(Ben Mena,2000) Ben Mena S. *Introduction aux méthodes multicritères d'aide à la décision*. Biotechnol.Agron.Soc.Environ. 4 (2), 83-93. 2000.

(Benatallah et *al.*,2003) Benatallah B., Sheng Q.Z. and Dumas M. *The Self-Serv Environment for Web Services Composition*. IEEE Internet Computing, 7(1), pp.40-48, January 2003.

(Benatallah et *al.*,2005) Benatallah B., Dijkman R., Dumas M. and Maamar Z. *Service Composition : Concepts, Techniques, Tools and Trends*. In Stojanovic Z., Dahanayake A., Eds. Service-Oriented Software Engineering: Challenges and Practices. Idea Group Inc (IGI), pp.48-66, 2005.

(Ben Achour & Ncube, 2000) Ben Achour C., Ncube C. *Engineering the PORE Method for COTS Selection and Implementation with the MAP Process Meta-Model*. Proceedings of REFSQ'2000, 6th International Workshop on Requirements Engineering Foundations of Software Quality, Stockholm, Sweden, 2000.

(Ben Hamida, 2008) Ben Hamida A., Le Mouël F., Frénot S. and Ben Ahmed M. *Une approche pour un chargement contextuel de services dans les environnements pervasifs*. Ingénierie des Systèmes d'Information (ISI) pp. 59-82. 2008.

(Bennasri, 2005) Bennasri I. *Une approche intentionnelle de représentation et de réalisation de la variabilité dans un système logiciel*. PhD. thesis, Université Paris1 Panthéon Sorbonne, 2005.

(Boehm et *al.*,1976) Boehm B.W, Brown J.R and Lipow M. *Quantitative evaluation of software quality*. In Proceedings of the 2nd international conference on Software engineering, pp.592 – 605. 1976.

(Boehm et *al.*,1978) Boehm B. W., Brown J. R., Kaspar H., Lipow M., McLeod G. and Merritt M. *Characteristics of Software Quality*. North Holland, 1978.

(Booth et *al.*,2004) Booth D., Haas H., Mccabe F., Newcomer E., Champion M., Ferris C., and Orchard D. *Web services architecture*. Technical report, W3C, Web Services Architecture Working Group, February 2004.

(Brans et *al.*,1984) Brans JP., Mareschal M. and Vincke P. PROMETHEE : A New Family of Outranking Methods In Multicriteria Analysis. In Oper. Research '84, éd. par J.P. Brans, North-Holland, 1984.

(Brown,1996) Brown A-W. *Component-based software engineering : selected papers from the Software Engineering Institute*. Los Alamitos, CA : IEEE Computer Society Press, September 1996.

(Buisson et *al.*,1987) Buisson J.C., Farreny H., Prade H. *Mise en oeuvre de techniques de raisonnement approché dans un système expert. L'exemple de DIABETO-III.* Dans : Innovation et Technologie en Biologie et Médecine, Vol. 8, N. spécial 2, p. 71-89, 1987.

(Bühne et *al.*, 2005) Bühne S., Lauenroth K., Pohl K. *Modelling Requirements Variability across Product Lines*. In: Atlee, Joanne, M. (Eds.) 13th IEEE Intl. Conference on Requirements Engineering. IEEE Computer Society, pp. 41-50, September 2005.

(Cardoso et *al.*,2004) Cardoso J., Sheth A., Miller J., Arnold J. and Kochut K. *Quality of service for workflows and web service processes*. Journal of Web Semantics, 2004.

(Caillet,2003) Caillet R. *Analyse multicritère : Étude et comparaison des méthodes existantes en vue d'une application en analyse de cycle de vie*. CIRANO, Montréal, août 2003.

(Castro et *al.*,2002) Castro J., Kolp J. and Mylopoulos J. *Towards requirements-driven information systems engineering : the Tropos project*. Info. Syst. 27, pp. 365-389, 2002.

(Casati&Shan,2001) Casati F., Shan M.C. *Models and languages for describing and discovering e-services (tutorial).* In SIGMODConference, pp. 626, USA, 2001.

(Cerami,2002) Cerami E. *Web Services Essentials*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2002.

(Chauvet,2002) Chauvet J-M. *Services WEB avec SOAP, WSDL, UDDI, ebXML*. Eds Eyrolles, Paris, 2002.

(Checkland&Holwell,1998) Checkland P. and Holwell S. *Information, Systems and Information Systems - Making Sense of the Field*. Wiley. 1998.

(Chen&Hwang,1992) Chen S.J. and Hwang C.L. *Fuzzy Multiple Attribute Decision Making : Methods and Applications*. Lecture Notes in Economics and Mathematical Systems, No. 375, Sringer-Verlag, Berlin, Germany, 1992.

(Chung et *al.*,2000) Chung L., Nixon B., Yu E. and Mylopoulos J. *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers. 2000.

(Chung,1991) Chung L. *Representing and utilization of Non-Functional Requirements for information system design*. In Proc. 3rd Int. Conf. Advanced Information Systems Engineering, CaiSE, 1991.

(Chung&Nixon,1995) Chung L. and Nixon A.N. *Dealing with Non-Functional Requirements: Three experimental studies of a process-oriented approach*. In Proc., 17th International Conference on Software Engineering (ICSE), Seattle, WA, U.S.A., pp. 25--37. Apr. 1995.

(Chung et *al.*,1996) Chung L., Nixon B.A. and Yu E. *Dealing with Change : An Approach Using Non-Functional Requirements*. Requirements Engineering Journal, 1(4), pp. 238. 1996.

(Chung&Subramanian,2001) Chung L. and Subramanian N. *Process-Oriented Metrics for Software Architecture Adaptability*. Proceedings of the Fifth IEEE International Symposium on Requirements Engineering, pp. 310. 2001.

(Clark et *al.*,1992) Clark D., Shenker S. and Zhang L. *Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism*. Proceedings of ACM SIGCOMM. pp. 14-26. 1992.

(Clements&Northrop, 2001) Clements P. and Northrop L. *Software Product Lines: Practices and Patterns*. Addison-Wesley Longman Reading, Boston, August 2001.

(Cruz,1995) Cruz R.L. *Quality of service guarantees in virtual circuit switched networks*. IEEE J. Select. Areas Commun. 13(6): 1048. 1995.

(Cysneiros et *al.*,2001) Cysneiros L.M., Leite J.C.S.P. and Neto J.S.M. *A Framework for Integrating Non Functional Requirements into Conceptual Models*. Requirements Engineering Journal -- Vol 6, pp.97-115, Issue 2 Apr. 2001.

(Comes et *al.*,2010) Comes D.E, Baraki H., Reichle R., Zapf M. and Geihs K. *Heuristic Approaches for QoS-Based Service Selection*. In ICSOC, pp.441-455, 2010.

(Chen et *al.*, 2003) Chen M.F., Tzeng G.H. and Ding C.G. *Fuzzy MCDM approach to select service provider*. In Proc. of the 12th IEEE International Conference on Fuzzy Systems, St. Louis, MO, USA, Vol. 1, pp. 25-28, May 2003.

(Dardenne et *al.*,1991) Dardenne A., Fickas S. and van Lamsweerde A. *Goal-Directed Concept Acquisition in Requirements Elicitation*. In Proc. IWSSD-6 - 6th Intl. Workshop on Software Specification and Design, Como, pp. 14- 21. 1991,.

(Dardenne et *al.*,1993) Dardenne A., van Lamsweerde A. and Fickas S. *Goal-Directed Requirements Acquisition*. Science of Computer Programming, Vol. 20, pp.3-50. 1993.

(Darimon&Van Lamsweerde,1996) Darimont R.and Van Lamsweerde A. *Formal Refinement Patterns for Goal-Driven Requirements Elaboration*. In Proc. FSE'4 - Fourth ACM SIGSOFT Symp. on the Foundations of Software Engineering, pp. 179-190, San Francisco, October 1996.

(Da Silva Santos et *al.*,2009) Da Silva Santos L. O. B., Guizzardi G., Pires L. F., Van Sinderen M. *From User Goals to Discovery and Composition*. Advances in Conceptual Modeling - Challenging Perspectives. Lecture Notes in Computer Science 5833/2009. Springer Verlag, Berlin, pp. 265-274, 2009.

(Davis,1993) Davis A. *Software Requirements: Objects, Functions and States*. Prentice Hall, 1993.

(Djebbi&Salinesi,2006) Djebbi O and Salinesi C. *Criteria for Comparing Requirements Variability Modeling Notations for Product Lines*. Fourth International Workshop on Comparative Evaluation in Requirements Engineering (CERE'06 - RE'06 Workshop).2006.

(Djebbi,2011) Djebbi O. *L'ingénierie des exigences par et pour les lignes de produits*. PhD. thesis, Université Paris1 Panthéon Sorbonne, 2011.

(Deora et *al.*,2003) Deora V., Shao J., Gray W. A. and Fiddian N. J. *A Quality of Service Management Framework Based on User Expectations*. ICSOC, pp.104-114. 2003.

(Dik,1989) Dik S.C. *The theory of functional grammar*. Foris Publications, Dodrecht, Pays-Bas, 1989.

(Dobson et *al.*,2005) Dobson G., Lock R. and Sommerville I. *QoSOnt : a QoS ontology for service-centric systems*. In Software Engineering and Advanced Applications. 31st EUROMICRO Conference, September 2005.

(Donzelli,2004) Donzelli P. *A goal-driven and agent-based requirements engineering framework*. In RE, pp.16–39, 2004.

(Dustdar&Schreiner,2005) Dustdar S., Schreiner W.A. *Survey on Web services composition*. In International Journal of Web and Grid Services, 1(1), 2005.

(Edme, 2004) Edme M-H. *Proposition pour la modélisation et le guidage des systèmes d'information multi-facettes*. Manifestation des Jeunes Chercheurs en Sciences et Technologies de l'Information et de la Communication, MAJECSTIC 2004.

(Edme, 2005) M-H Edme. *Proposition pour la modélisation intentionnelle et le guidage de l\'usage des systèmes d\'information*. PhD. thesis, Université Paris1 Panthéon Sorbonne 2005.

(Fang et *al.*,2004) Fang D.P., Vie X.Y. and Li H. *Factors analysis-based studies on construction workplace safety management in China*. International Journal of Project Management, 22(1), pp.43-49. 2004.

(Fante,2004) Fante S. *Goal-Oriented Requirements Engineering : tecniche Numeriche di Analisi Top-down and Bottom-up*. Master's thesis, Department of Information and communication Technology University of Trento, 2004.

(Fedosseev,2004) Fedosseev P. *Composition of Web Services and QoS aspects*. Seminar: Data Communication and Distributed Systems. (Disponible sur http://www.nets.rwthaachen.de/content/teaching/seminars/sub/2003_2004_ws_docs/WebServices.pdf), WS 2003/2004.

(Fernandez,2000) Fernandez A. *Les nouveaux tableaux de bord des décideurs*. Paris, Éditions d'organisation, 2000.

(Fenton&Pfleeger,2004) Fenton N.E. and Pfleeger S.L. *Software Metrics : A Rigorous & Practical Approach* (2nd ed.Revisited Printing). London: International Thomson Computer Press. 2004.

(Fillmore,1968) Fillmore C.J. *The case for case, in Universals in linguistic theory*. Holt, Rinehart and Winston, Inc, E.Bach/R.T.Harms (eds), 1968.

(FIPA,2002) FIPA- Foundation for Intelligence Physical Agents. *Quality of Service Ontology Specification*. Switzerland, November 2002 (Disponible sur http://www.fipa.org/specs/fipa00094/XC00094.html#_Toc23835542).

(Fishburn,1967) Fishburn P.C. *Additive Utilities with Incomplete Product Set : Applications to Priorities and Assignments*. Operations Research Society of America (ORSA) Publication, Baltimore, MD, 1967.

(Fisher et *al.*,1991) Fisher G., Henninger S. and Redmiles D. *Intertwining query construction and relevance evaluation*. Conference on Human Factors in Computing Systems, Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology, pp.55-62, United States, 1991.

(Fouquet,2000) Fouquet B. *Gestion de la qualité de service*. Eds Eyrolles, 2000.

(Fremantle et *al.*,2002) Fremantle P., Weerawarana S. and Khalaf R. *Enterprise services*. Communication of the ACM, 45(10), 2002.

(Giorgini et *al.*,2004) Giorgini P., Mylopoulos J., Nicchiarelli E. and Sebastiani R. *Formal reasoning techniques for goal models*. Journal on Data Semantics, Springer, 2004.

(Glinz,2005) Glinz M. *Rethinking the Notion of Non-Functional Requirements*. In Proceedings of the 3rd World Congress for Software Quality (3WCSQ 2005), pp.55-64, Germany, 2005.

(Glinz,2007) Glinz M. *On Non-Functional Requirements*. In 15th IEEE International Requirements Engineering Conference (RE), pp.21-26, India, October 2007.

(Glinz,2008) Glinz M. *A Risk-Based, Value-Oriented Approach to Quality Requirements*. IEEE Software,2008.

(Guiffrida&Nagi,1998) Guiffrida A.L. and Nagi R. *Fuzzy set theory applications in production management research : a literature survey*. Journal of intelligent manufacturing, 9(1), pp. 39-56. 1998.

(Guzélian,2007) Guzélian G. *Conception de systèmes d'information : une approche orientée service*. PhD thesis, Université de Paul Cézanne d'Aix-Marseille III, 2007.

(Giallonardo&Zimeo,2007) Giallonardo E. & Zimeo E. *More Semantics in QoS Matching*. In Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications (SOCA'07). 2007.

(Gamma et *al.*,1997) Gamma E., Helm R., Johnson R. and Vlissides J .*Design Patterns, Elements of Reusable Object-Oriented Software*. ISBN 0-201- 63361-2 , 1997, Addison-Wesley.

(Hall&Fenton,1997) Hall T. and Fenton N. *Implementing Effective Software Metrics Programs*. IEEE Software, pp. 55 – 65.1997.

(Harel&Naamad,1996) Harel D. and Naamad A. *The statemate semantics of statecharts*. ACM Transactions on Software Engineering and Methodology, 5(4), pp. 293-333, 1996.

(Herssens et *al.*,2008) Herssens C., Jureta I.J. and Faulkner S. *Capturing and Using QoS Relationships to Improve Service Selection*. In proceeding of CAISE, France 2008.

(Html,2009) Site de HTML : http://www.w3.org/Protocols/

(Huhns&Singh,2005) Huhns M.N. and Singh M.P. *Service-Oriented Computing : Key Concepts and Principles*. IEEE Internet Computing, 9(1), pp. 75-81, 2005.

(Hwang&Yoon,1981) Hwang C.-L. and Yoon K. *Multiple Attribute Decision Making : Methods and Applications*. Springer, Berlin, 1981.

(Huang et *al.*,2007) Huang C.L., Lo C.C., Chao K.M. and Younas M. *Reaching consensus: A moderated fuzzy web services discovery method*. Information and Software Technology, Vol. 48, Iss. 6, pp.410-423, June 2007.

(IBM,2007) IBM. Spécification BPEL4WS. Voir le site http://www.ibm.com/developerworks/library/specification/ws-bpel/

(Iacovelli,2009) Iacovelli A. *Services méthodologiques : concepts, architecture et plateforme*. Actes du XXVII° congrès INFORSID, Toulouse, mai 2009.

(ISO/IEC 9126,2001) ISO/IEC 9126. *Software engineering- Product quality- Part 1: Quality model*. International Organization for Standardization. 2001.

(ISO 9000,2000) ISO 9000:2000. *Quality Management Systems- Fundamentals and Vocabulary*. International Organization for Standardization. 2000.

(ISO/IEC 9001,2008) ISO/IEC 9001:2008. *Quality Management Systems- Fundamentals and Vocabulary*. International Organization for Standardization. 2008.

(ISO/IEC 25000,2005) ISO/IEC 25000:2005. *Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE*. 2005.

(ISO/IEC 25010,2011) ISO/IEC 25010:2011. *SQuaRE : Systems and software engineering -- Systems and software Quality Requirements and Evaluation -- System and software quality models*.(Disponible sur http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35733). 2011.

(IEEE,1990) IEEE. *Standard Glossary of Software Engineering Terminology*. IEEE Standard 610.12-1990.

(IEEE,1993) IEEE. *IEEE Recommended Practice for Software Requirements Specifications*. IEEE Standard 830-1993.

(ISIC Rev. 4.0,2008) ISIC Rev. 4.0. *International Standard Industrial Classification of All Economic Activities. Revision 4*. Statistical papers. disponible sur : http://www.escwa.un.org/divisions/div_editor/Download.asp?table_name=other&field_name=ID&FileID=1020. United Nations, New York, 2008.

(ITU,1994) ITU-T E-800. *Terms and definitions related to quality of service and network performance including dependability*. Telecommunication Standardization Sector Of International Telecommunication Union(ITU), August, 1994.

(Jackson,1995) Jackson M. *Software Requirements and Specifications – A Lexicon of Practice, Principles and Pejudices*. ACM Press, Addison-Wesley, 1995.

(Jaeger et *al.*,2005) Jaeger M. C., M¨uhl G., and Golze S. *QoS-aware Composition of Web Services: An Evaluation of Selection Algorithms*. In Proceedings of the Confederated International Conferences CoopIS, DOA, and ODBASE 2005 (OTM'05), Agia Napa, Cyprus,

ser. Lecture Notes in Computer Science (LNCS), vol. 3760, pp. 646–661,Springer, November 2005.

(Jureta et *al.*,2006) Jureta I.J., Faulkner S. and Schobbens P-Y. *A more expressive softgoal conceptualization for quality requirements analysis*. In ER 2006.

(Jureta et *al.*,2007) Jureta I.J., Faulkner S., Achbany Y. and Saerens M. *Dynamic Web Service Composition within a Service-Oriented Architecture*. In ICWS 2007.

(Jureta et *al.*,2008) Jureta I.J., Herssens C. and Faulkner S. *A comprehensive quality model for service-oriented systems*. In Software quality journal, 2008.

(Kaabi et *al.*,2004) Kaabi R. S., Souveyet C. and Rolland C. *Eliciting service composition in a goal driven manner*. Int. Conf. on Service Oriented Computing, ICSOC'04, New York, Dec 2004.

(Kaabi,2007) Kaabi R. *Une approche méthodologique pour le développement d'applications à base de services*. PhD. thesis, Université Paris1 Panthéon Sorbonne, 2007.

(Karloff,1991) Karloff H. *Linear Programming : (Progress In Theoretical Computer Science)*. Birkhauser, 1991.

(Kartam et *al.*,2004) Kartam N. A., Flood I. and Koushki P. *Construction safety in Kuwait : issues, procedures, problems and recommendations*. Safety Science, 36(3), 163–184. 2004.

(Keller et *al.*,1990) Keller S.E., Kahn L.G. and Panara R.B. *Specifying Software Quality Requirements with Metrics*. In Thayer, R.H.; Dorfman. M.: System and Software Requirements Engineering, IEEE Computer Society Press, pp. 145, Washington, 1990.

(Kellert&Toumani,2003) Kellert P. and Toumani F. *Les services Web sémantiques*. Research Report LIMOS/RR, 2003 - revue-i3.org. 2003.

(Khalaf&Leymann,2003) Khalaf R. and Leymann F. *On Web Services Aggregation*. TES, LNCS 2819, Spring-Verlag Berlin Heidelberg, pp.1-13, 2003.

(Kobielus,1997) Kobielus J.G. *Workflow Strategies*. IDG Books WorldWide, 1997.

(Köksalan&Zionts,2000) Köksalan M. and Zionts S. *Multiple Criteria Decision Making in the new millenium*. In Proceedings of the 15th Intl Conference on MCDM, Turkey, 2000. Eds Springer-Verlag. 2000.

(Kritikos& Plexousakis,2007) Kritikos K. & Plexousakis D. *Extending OWL for QoS-based Web Service Description and Discovery*. In Proceedings of the Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2'07), Busan, South Korea. 2007.

(Lamsveerde,2001) Van Lamsveerde A. *Goal-Oriented Requirements Engineering : A Guided Tour*. In Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE'01), Canada, August 2001.

(Lee et *al.*,2003) Lee K., Jeon J., Lee W., Jeong S-H. and Park S-W. *QoS for web services : Requirements and possible approaches*. Technical report, W3C, Web Services Architecture Working Group, November 2003.

(Liaudanskiene et *al.*,2009) Liaudanskiene R., Ustinovicius L. and Bogdanovicius A. *Evaluation of Construction Process Safety Solutions Using the TOPSIS Method*. In economics of Engineering Decisions. ISSN 1392-2785. 2009.

(Liu&Yu,2004) Liu L. and Yu E. *Designing information systems in social context : a goal and scenario modeling aproach*. Info. Syst. 29, pp. 187-203, 2004.

(Li et *al.*,2011) LI F., HE Y., HU W., WU L. and WEN P. *Web Service Selection Based on Fuzzy QoS Attributes*. in Journal of Computational Information Systems, Vol. 7, No 1, pp.198- 205, 2011.

(Letier&Lamsweerde,2004) Letier E. and Van Lamsweerde A. *Reasoning about Partial Goal Satisfaction for Requirements and Design Engineering*. In Proceedings of the 12th ACM SIGSOFT 12th international symposium on Foundations of software engineering, pp.53–62, 2004.

(Lapouchnian et *al.*,2007) Lapouchnian A., Yu Y. and Mylopoulos J. *Requirements-Driven Design and Configuration Management of Business Processes*. In Proceeding BPM'07 Proceedings of the 5th international conference on Business process management.

(Lutz, 2000) Lutz R. R. *Extending the product family approach to support safe reuse*. Journal of Systems and Software. 2000.

(Ma et *al.*,2009) Ma W., Liu L., Xie H., Zhang H. and Yin J. *Preference Model Driven Services Selection*. In 21st International Conference on Advanced Information Systems Engineering (CAISE'09), pp. 216-230, 2009.

(MacNaugthon&Yamada,1960) MacNaughton R. and Yamada. *Regular expressions and state graphs for automata*. IEEE transactions on electronic computers, EC-9, P 39-47, 1960.

(Mahbub&Spanoudakis,2004) Mahbub K. and Spanoudakis G. *A framework for requirements monitoring of service based systems*. In: ICSOC'04: In Proc. of the 2nd Int. Conf. on Service Oriented Computing.2004.

(Mani&Nagarajian,2002) Mani A. and Nagarajian A. *Web services : Understanding quality of service for Web services*. IBM Developer Works, January 2002, (disponible sur http://www-106.ibm.com/developerworks/library/ws-quality.html).

(Martn-Daz et *al.*,2003) Martn-Daz O., Corts A. R., Durn A., Benavides D. and Toro M. *Automating the procurement of web services*. In Service-Oriented Computing (ICSOC), pp.91–103, LNCS 2910, Springer, 2003.

(Masud&Hwang,1981) Masud AS. and Hwang CL. Interactive sequential goal programming. J. Oper. Res. Soc. 32 (5) p. 391–400. 1981.

(Maximilien&Singh,2002) Maximilien E. M. and Singh M. P. *Conceptual model of Web service reputation*. ACM SIGMOD Record, 31(4), December 2002.

(Maximilien&Singh,2004a) Maximilien E. M. and Singh M. P. *Toward autonomic services trust and selection*. In Proceedings of the International Conference on Service-Oriented Computing (ICSOC'04), 2004.

(Maximilien&Singh,2004b) Maximilien E. M. and Singh M. P. *A Framework and Ontology for Dynamic Web Services Selection.* IEEE Internet Computing, 8(5), pp.84–93, September 2004.

(McCall&Matsumoto,1980) McCall J.A. and Matsumoto M.T. *Software Quality Metrics Enhancements*. General Electric Company, RADC-TR-80-109, Two volumes, Rome Air Development Center, April 1980.

(Medjahed et *al.*,2003) Medjahed B., Benatallah B., Bouguettaya A., Ngu A.H.H., Elmagarmid H.K. *Business-to-business interactions: issues and enabling technologies*. The VLDB Journal. 2003.

(Mendling&Hafner,2008) Mendling J. and Hafner M. *From WS-CDL choreography to BPEL process orchestration*. Journal of Enterprise Information Management, Vol. 21 Iss: 5, pp.525 – 542. 2008.

(Merrick&Harrald,2007) Merrick j.R.W and Harrald J.R. *Making Decision About Safety in US Ports and Waterways*. INTERFACES, 2007.

(Miller&Starr,1969) Miller D.W. adn Starr M.K. *Executive Decisions and Operations Research*. Prentice-Hall, Inc.,Englewood Cliffs, NJ, 1969.

(Mirbel&Crescenzo,2009) Mirbel I. and Crescenzo P. *Des besoins des utilisateurs à la recherche de services web : une approche sémantique guide par les intentions.* Rapport de recherche ISRN I3S/RR-2009-12-FR, 2009.

(Mostow,1985) Mostow J. *Towards better models of the design process*. AI Magazine, vol.6, no. 1, pp. 44-57, Spring 1985.

(Mohabbati et *al.*2011) Mohabbati B., Hatala M., Gaˇsevi´c D., Asadi M. and Boˇskovi´c M. Development *and configuration of service-oriented systems families*. In: Proceedings of the 2011 ACM Symposium on Applied Computing. SAC '11, New York, NY, USA, ACM 1606–1613. 2011.

(Munda,1995) Munda G. Multicriteria evaluation in a fuzzy environment, contributions to economics Series, Physica-Verlag, Heidelberg.

(Mylopoulos et *al.*,1992) Mylopoulos J., Chung L. and Nixon. B. *Representing and Using Non-Functional Requirements : A Process-Oriented Approach*. In IEEE Transactions on Software Engineering, 18(6), pp.483-497, June 1992.

(Mylopoulos&Castro,2000) Mylopoulos J. and Castro J. *TROPOS : A framework for requirements driven software development. Information systems engineering: state of the art and research themes.* Lecture Notes in Computer Science, Springer-Verlag, 2000.

(Nilsson,1971) Nilsson N.J. *Problem Solving Methods in Artificial Intelligence*. McGraw Hill, 1971.

(OMG, 2008) Object Management Group. *UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms*. Version 1.1, formal/2008-04-05, April 2008.

(Ommering&Bosch, 2002) Ommering R.V. and Bosch J. *Widening the Scope of Software Product Lines - From Variation To Composition*. In Proceedings of the Second Software Product Line Conference. 2002.

(Ommering, 2002) Ommering R.V. *Building product populations with software components*. In Proceeding ICSE '02 Proceedings of the 24th International Conference on Software Engineering. 2002.

(Orriëns et *al.*,2003) Orriëns B. Yang J. and Papazoglou M.P. *Model driven service composition*. In ICSOC 2003.

(O'Sullivan et *al.*,2002) O'Sullivan J., Edmond D., and Hofstede.A.T. *What's in a Service? Towards Accurate Description of Non-Functional Service Properties*. In Distributed Parallel Databases, 2002.

(Pallos,2001) Pallos M. *Service Oriented Architecture: A Primer*. EAI Journal, December 2001.

(Papaioannou et *al.*,2006) Papaioannou I.V., Tsesmetzis D.T., Roussaki I.G. and Anagnostou M.E. *A QoS ontology language for web services*. In 20th International Conference on Advanced Information Networking and Applications, April 2006.

(Papazoglou&Georgakapoulos,2003) Papazoglou M.P. and Georgakapoulos G. *Introduction to a Special Issue on Service Oriented Computing*. Communications of the ACM, 46(10), October 2003.

(Papazoglou,2005) Papazoglou M-P. *Extending the Service Oriented Architecture*. Business Integration Journal, February 2005.

(Papazoglou et *al.*,2008) Papazoglou M-P., Traverso P., Dustdar S. and Leymann F. *Service-Oriented Computing : A Research Roadmap*. Int. J. Cooperative Inf. Syst. 17(2), pp.223-255, 2008.

(Papazoglou,2007) Papazoglou M.P. *Web Services : Principles and Technology*. Pearson - Prentice Hall, 782 pages, July 2007.

(Peltz,2003) Peltz C. *Web Services Orchestration and Choreography*. IEEE Computer, 36(10), pp.46-52, 2003.

(Penserini&Mylopoulos,2005) Penserini L. and Mylopoulos J. *Design Matters for Semantic Web Services*. ITC-IRST, Technical report: T05-04-03, April 2005.

(Penserini et *al.*,2006a) Penserini L., Perini A., Susi A. and Mylopoulos J. *From Stakeholder needs to service requirements specifications*. Technical Report, ITC-IRST, Automated Reasoning Systems, 2006.

(Penserini et *al.*,2006b) Penserini L., Perini A., Susi A.and Mylopoulos J. *From Stakeholder intentions to software agent implementations*. CaiSE'06, LNCS 4001, pp. 465-479, 2006.

(Pfister&Szyperski,1996) Pfister C. and Szyperski C. *Why objects are not enough*. In Proceedings of International Component Users Conference, SIGS, Germany 1996.

(Piccinelli et *al.*,2003) Piccinelli G., Emmerich W., Williams S-L. and Stearns M. *A Model-Driven Architecture for Electronic Service Management Systems*. In the Proc. of the first International Conference Service-Oriented Computing - ICSOC, Italy, December 2003.

(Pinedo,2001) Pinedo M.L. *Scheduling : Theory, Algorithms and Systems* (2nd Edition). Prentice Hall. 2001.

(Prat,1997) Prat N. *Goal formalisation and classification for requirements engineering*. In Proceedings of the Third International Workshop on Requirements Engineering: Foundations of Software Quality REFSQ'97, Barcelona, June 1997.

(Python,2009) Site de Python Programming Language : http://www.python.org/

(Ragland,1995) Ragland B. *Measure, Metric, or Indicator : What's the Difference?* http://www.stsc.hill.af.mil/crosstalk/1995/03/Measure.asp

(Ralyté,2001) Ralyté J. Ingénierie des méthodes à base de composants, Thèse de doctorat en informatique, Université Paris 1, 2001.

(Ran,2003) Ran S. *A Model for Web Services Discovery with QoS*. ACM SIGecom Exchanges, 4(1), March 2003.

(Robertson&Robertson,1999) Robertson S. and Robertson J. *Mastering the requirements process*. Addison-wesley, 1999.

(Rohleder et *al.*,2009) Rohleder C., Marhold C., Salinesi C. and Doerr J. *Quality Data Model and Quality Control in the Product Lifecycle Management*. International Conference on Product Lifecycle Management (ICPLM), Bath, Royaume Uni, July 2009.

(Rohleder,2009) Rohleder C. *Representation of software variability requirements*. PhD. thesis, Université Paris1 Panthéon Sorbonne, 2009.

(Rolland et *al.*,2010) Rolland C., Kirsch-Pinheiro M., Souveyet C. An *Intentional Approach to Service Engineering*. IEEE T. Services Computing 3(4): 292-305. 2010.

(Rolland et *al.*,1998a) Rolland C., Ben Achour C., Cauvet C., Ralyte J., Sutcliffe A., Maiden N. A. M., Jarke M., Haumer P., Pohl K., Dubois E. and Heymans P. *A Proposal for a Scenario Classification Framework*. In Requirements Engineering Journal (REJ), 3(1), pp. 23-47, 1998.

(Rolland et *al.*,1998b) Rolland C., Souveyet C. and Salinesi C. *Guiding Goal Modelling using Scenarios*. In IEEE Trans. On Software Engineering (Special Issue on Scenario Management), 24(12), pp.1055-1071,1998.

(Rolland, 1999) Rolland C. *Requirements Engineering for COTS based Systems*. Journal of Information and Software Technology (JIST), Elsevier, 41, pp. 985 - 990, 1999.

(Rolland&Prakash,2000) Rolland C. and Prakash N. *Bridging the gap between Organizational needs and ERP functionality*. Requirements Engineering Journal, 2000.

(Rolland&Prakash,2001) Rolland C. and Prakash N. *Matching ERP System Functionality to Customer Requirements*. In Proceedings of the 5th International Symposium on Requirements Engineering (RE'01), Toronto, Canada, pp. 66-75, 2001.

(Rolland,2007) Rolland C. *Capturing System Intentionality with Maps*. Conceptual Modelling in Information Systems Engineering, Springer-Verlag, Berlin heidelberg, pp. 141 – 158,Germany, 2007.

(Rolland,2003) Rolland C. *Reasoning with Goals to Engineer Requirements*. In Proceeding 5th International Conference on Entreprise Information Systems, 2003.

(Rolland et *al.*,2004) Rolland C., Salinesi C. and Etien A. *Eliciting Gaps in Requirements Change*. In Requirements Engineering Journal (REJ), 9:1, pp. 1 - 15, 2004.

(Rolland et *al.*,2008) Rolland C., Souveyet C. and Kraeim N. *An intentional view of service-oriented computing*. Revue Ingénierie des Systèmes d\'Information (ISI), RSTI (Revue des Sciences et Technologies de l\'Information)- ISI – 13 - n°1/2008, Eds Hermès, 13(1), pp. 107 - 137, France, 2008.

(Rolland et *al.*,2007) Rolland C., Kaabi R. S. and Kraeim N. *On ISOA: Intentional Services Oriented Architecture*. International Conference on Advanced information Systems Engineering (CAISE), Springer-Verlag, Norway, June 2007.

(Rolland&Kaabi,2007) Rolland C. and Kaabi R. S. *An Intentional Perspective to Service Modeling and Discovery*. International Computer Software and Applications Conference (COMPSAC), pp. 455 – 460, China, July 2007.

(Roman,1985) Roman G-C. *A taxonomie of current issues in requirements engineering*. IEEE computer, Vol. 18, no.4, pp.14-23, April 1985.

(Roy&Bouyssou,1993) Roy B. and Bouyssou D. *Aide multicritères à la décision : méthodes et cas*. Edition Economica. 1993.

(Roy,1968) Roy B. Classement *et choix en présence de points de vue multiples (la méthode Electre)*. Rev. Fr. Inf. Rech.Opér. 2 (8) p. 57–75. 1968

(Roy,1992) Roy, B. *Sciences de la décision ou sciences d'aide à la décision ?* Revue Internationale de Systémique, vol. 6, nº 5, 1992.

(Ruiz et *al.*,2003) Ruiz F., Genero M., García F., Piattini M., Calero C. *A proposal of a software measurement ontology*. 32th JAIIO – 2003, Jornadas Argentinas de Informática e Investigación Operativa Argentine, Conference on Computer Sciences and Operational Research, Simposio Argentino de Ingeniería de Software, http://www.frcu.utn.edu.ar/deptos/depto_3/32JAIIO/asse/asse_02.pdf, 1-5 sept. 2003.

(Saaty,1977) Saaty T.L. *A scaling method for priorities in hierarchical structures*. Journal of Mathematical Psychology, 15:234-281, 1977.

(Saaty,1990) Saaty T.L. *How to make a decision : The Analytic Hierarchy Process*. European Journal of Operational Research, 48:9-26, 1990.

(Saaty&Vargas,2001) Saaty T. and Vargas L. *Models, Methods, Concepts and Applications of the Analytic Hierarchy Process*. Boston: Kluwer. 2001

(Saaty et *al.*,2007) Saaty T.L., Peniwati K. and Shang J.S. *The analytic hierarchy process and human resource allocation : Half the story*. Mathematical and Computer Modelling, 46(7-8):1041 - 1053, October 2007.

(Salinesi&Rolland,2003) Salinesi C. and Rolland C. *Fitting Business Models to Software Functionality : Exploring the Fitness Relationship*. In Proceedings of the 15th Conference on Advanced Information Systems Engineering,(CAISE'03), Springer Verlag (pub), 2003.

(Sawyer at *al.*,2005) Sawyer P., Hutchinson J., Walkerdine J. and Sommerville I. Faceted *Service Specification*. In Proceedings SOCCER (Service-Oriented Computing: Consequences for Engineering Requirements) Workshop, at RE'05 conference, France, august 2005.

(Schinas,2007) Schinas O. *Examining the use and the application of Multi-Criteria Decision Making Techniques in Safety Assessment*. in international symposium on maritime safety, security&environmental protection. Athens, September 2007.

(Schmidt et *al.*,1998) Schmidt D.C., Levine D.L. and Mungee S. *The Design and Performance of Real-Time Object Request Brokers*. Computer Communications, v.21, pp. 294—324. 1998.

(Schärlig,1985) Schärlig A. Décider sur plusieurs critères, panorama de l'aide à la décision multicritère. Lausanne, Suisse : Presses polytechniques et universitaires romandes, pp. 304. 1985.

(Sebastiani et *al.*,2004) Sebastiani R., Giorgini P. and Mylopoulos J. *Simple and minimum-cost satisfiability for goal models*. In Proceedings of the 16th Conference On Advanced Information Systems Engineering (CAiSE'04). LNCS, Springer, 2004.

(SeCSE,2009) Site du projet SeCSE: http://www.secse-project.eu/

(Sheng et *al.*,2002) Sheng Q. Z., Benatallah B., Dumas M. and Mak E. *SELF-SERV: A Platform for Rapid Composition of Web Services in a Peer-to-Peer Environment*. In Proceedings of the 28th VLDB conference, China. August 2002.

(Siqueira&Cahill,1998) Siqueira F. and Cahill V. *Quartz : Supporting QoSConstrained Services in Heterogeneous Environments*. RTSS'98 Proceedings (work in progress), November 1998.

(Simon,1981) Simon H.A. *The science of the Artificial*. Second Edition, Cambridge, MA: The MIT Process. 1981.

(Singh&Bilgin,2004) Singh M. P. and Bilgin S. *A DAML-based repository for QoS -aware semantic web service selection*. In IEEE International Conference on Web Services (ICWS 2004), 2004.

(Soap,2007) Site de SOAP : http : //www.w3.org/TR/soap/

(Sommerville,2004) Sommerville I. *Software Engineering*. Seventh Edition. Pearson Education. 2004.

(Stollberg&Norton,2007) Stollberg M. and Norton B.*A Refined Goal Model for Semantic Web Services*. In Proceedings of the 2nd International Conference on Internet and Web Applications and Services (ICIW 2007), Mauritius, 2007.

(Supakkul&Chung,2004) Supakkul S. and Chung L. *Integrating FRs and NFRs : A Use Case and Goal Driven Approach*. In Proc. SERA 2004.

(Sun et *al.*,2011) Sun Q., Wang S., Zou H. and Yang F. *QSSA: A QoS-aware Service Selection Approach*. In International Journal of Web and Grid Services (May 2011), Vol. 7, No 2, pp .147-169, 2011.

(Sumra&Arulazi,2003) Sumra R. and Arulazi D. *Quality of Service for Web Services-Demystification, Limitations, and Best Practices*. (Disponible sur http://www.developer.com/services/article.php/2027911), March 2003.

(Szyperski,1997) Szyperski C. *Component software : beyond object-oriented programming*. ACM Press Harlow, England Reading, Mass : Addison- Wesley, 1997.

(Tarski,2008) Tarski A. *Introduction à la logique*. Fac-similé de l'édition de 1969, Paris, Gauthiers-Villars - Louvain, E. Nauwelaerts. 2008.

(Teije et *al.*,2004) Teije A.T, Harmelen F.V. and Wielinga B. *Configuration of Web Services as Parametric Design*. Proceedings of the Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW'04).2004.

(Tian et *al.*,2003) Tian M., Gramm A., Naumowicz T., Ritter H. and Schiller J. *A concept for QoS integration in web services*. In 1st Web Services Quality Workshop (WQW2003) at WISE, 2003.

(Tille,2000) Tille M. *Choix de variantes d'infrastructure routières : méthodes multicritères*. PhD NO 2294 (2000). École polytechnique fédérale lausanne. 2000.

(Thayer&Dorfman,1990) Thayer R. and Dorfman M. *System and software requirements engineering*. IEEE computer society press, 1990.

(Thompson& Heimdahl, 2002) Thompson J. and Heimdahl M. *Structuring Product Family Requirements for n-Dimensional and Hierarchical Product Lines*. In Requirements Engineering Journal. 2002.

(Triantaphyllou et *al.*,1998) Triantaphyllou E., Shu B., Nieto Sanchez S. and Ray T. *Multi-criteria decision making : An Operations Research Approach*. Encyclopedia of Electrical and Electronics Engineering, (J.G. Webster, Ed.), John Wiley & Sons, New York, NY, Vol. 15, pp. 175-186, 1998.

(Triantaphyllou,2000) Triantaphyllou E. *Multi-criteria decision making methods : a comparative study*. Dordrechet : Kluwer Academic Publishers, 290. 2000.

(Uddi,2004) Site de UDDI : http ://www.uddi.org/ par OASIS Open.

(Van der Aalst,2003) W.M.P. Van der Aalst. *Don't go with the flow: Web services composition standards exposed*. Journal of IEEE Intelligent Systems.2003.

(Van Lamsweerde,2001) Van Lamsweerde A. *Goal-Oriented Requirements Engineering : A Guided Tour*. Proc. 5th IEEE International Symposium on Requirements Engineering (RE'01), Toronto, Canada, August 2001.

(Van Lamsweerde,2000) Van Lamsweerde A. *Requirements Engineering in the Year 00 : A Research Perspective*. 22nd International Conference on Software Engineering (ICSE'2000), Limerick, Ireland, June 2000.

(Verma et *al.*,2004a) Verma K., Akkiraju R., Goodwin R., Doshi P., Lee J. *On Accommodating Inter Service Dependencies in Web Process Flow Composition*. In AAAI Spring Symposium PP: 37-43 on Semantic Web Services. 2004.

(Verma et *al.*, 2004b) Verma K., Aggarwal R., Miller J., Sheth A. *METEOR-S – An Environment for creating Semantic Web Processes*. In VLDB Journal, 2004.

(Verma et *al.*, 2005) Verma K., Gomadam K., Sheth A.P., Miller J.A. and Wu Z. *The METEOR-S Approach for Configuring and Executing Dynamic Web Processes*. Technical Report. 2005.

(Vincke,1989) Vincke P. L'Aide Multicritère à la Décision, Editions de l'Université de Bruxelles-Editions Ellipses, Bruxelles. 1989.

(Walkerdine at *al.*,2007) Walkerdine J., Hutchinson J., Sawyer P., Dobson G. and Onditi V. *A Faceted approach to service specification*. In Proceedings of 2nd Int'l Conf. on Internet and Web Applications and Services (ICIW'07), Mauritius, 2007

(Wang et *al.*,2006) Wang X., Vitvar T., Kerrigan M. and Toma I. *A QoS-Aware Selection Model for Semantic Web Services*. In ICSOC. 2006.

(Weerawarana et *al.*,2005) Weerawarana S., Curbera F., Leymann F., Storey T.and Ferguson D.F. *Web services platform Architecture*. Prentice Hall, 2005.

(WordNet,2009) WordNet. Disponible sur http://wordnet.princeton.edu/

(Wsdl,2001) WSDL. Disponible sur http : //www.w3.org/TR/wsdl/

(W3C,2003) W3C. *QoS for Web Services : Requirements and Possible Approaches*. 2003. (Disponible sur http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/).

(W3C,2004) W3C. *Web Services Architecture*. W3C Working Group Note 11 February 2004. (Disponible sur http://www.w3.org/TR/ws-arch/).

(Xiong&Fan,2007) Xiong P. and Fan Y. *QoS-aware Web Service Selection by a Synthetic Weight*. In Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge. Vol. 03, pp. 632-637. 2007.

(Xiong et *al.*,2008) Xiong P., Fan Y. and Zhou M. *QoS-Aware Web Service Configuration*. In IEEE Transactions on Systems Man and Cybernetics Part A Systems and Humans. 2008.

(Yamamoto&Saeki,2008) Yamamoto K. and Saeki M. *Attributed Goal-Oriented Analysis Method for Selecting Alternatives of Software Requirements*. IEICE Transactions 91-D(4): pp.921-932. 2008.

(Yu,1997) Yu E. *Towards modeling and reasoning support for early requirements engineering*. In Proc. of the IEEE Int. Symposium on RE, 1997.

(Yu et *al.*,1995) Yu E., Du Bois P., Dubois E. and Mylopoulos J. *From Organization Models to System Requirements - A `Cooperating Agents' Approach*. In Proc. 3rd Int. Conf. on Cooperative Information Systems (CoopIS-95), Vienna, Austria, pp. 194-204, May 1995.

(Yu,1995) Yu E. *Modeling Strategic Relationships for Process Reengineering*. Ph. D. Thesis, Department of Computer Science, University of Toronto, Canada (1995).

(Yue,1987) Yue K. *What Does It Mean to Say that a Specification is Complete?* In Proc. Fourth International Workshop on Software Specification and Design (IWSSD-4), Monterey, USA, 1987.

(Yu et *al.*,2007) Yu T., Zhang Y. and Lin K.-J. *Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints*. ACM Trans. Web, vol. 1, no. 1, p. 6, 2007.

(Zachos et *al.*,2006) Zachos K., Zhu X., Maiden N and Jones S. *Seamlessly integrating service discovery into UML requirements processes*. In Proceedings of Int'l workshop on Service-Oriented Software Engineering (SoSE'2006), ACM Press, china, 2006.

(Zachos et *al.*,2007) Zachos K., Maiden N, Jones S. and Zhu X. *Discovering Web Services To Specify More Complete System Requirements*. In Proceedings of CAiSE'07, 19th Conference on Advanced Information System Engineering, Norway, June 2007.

(Zachos et *al.*,2008a) Zachos K., Maiden N.A.M. and Howells-Morris R. *Discovering Web Services to Improve Requirements Specifications : Does It Help?* In Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'08), France, June 2008.

(Zachos et *al.*,2008b) Zachos K., Dobson G. and Sawyer P. *Ontology-aided Translation in the Comparison of Candidate Service Quality*. In Proceedings of SOCCER- Service-Oriented Computing Consequences for Requirements Engineering, workshop at RE'08, Spain, September 2008.

(Zeng et *al.*,2003) Zeng L., Benatallah B., Dumas M., Kalagnanam J. and Sheng Q.Z. *Quality driven web services composition*. In Proceedings of the International World Wide Web Conference (WWW'03), 2003.

(Zimmermann et *al.*,2004) Zimmermann O., Krogdahl P. and Gee C. *Elements of Service-Oriented Analysis and Design*. (Disponible sur http://www-128.ibm.com/developerworks/webservices/library/ws-soad1/), June 2004.

(Zeleny,1982) Zeleny M. *Multiple criteria decision making*. McGraw-Hill, Columbia University. 1982.

(Zavadskas,1986) Zavadskas, E. K. *The method of ranking of construction-technological alternatives on the basis of the distance from the ideal solution*. In: New contruction technology for buildings and structures. Leningrad, 52-57 (in Russian). 1986.

(Zhang et *al.*,2009) Zhang Y., Huang H., Yang D. and Zhang H. *A hierarchical and chord-based semantic service discovery System in the universal network.* International Journal of Innovative Computing, Information and Control, Volume 5, Number 11(A), November 2009.

(Zimmermann,1991) Zimmermann H.-J. *Fuzzy Set Theory and Its Applications*. Kluwer Academic Publishers, Second Edition, Boston, MA, 1991.

(Zoukar, 2005) Zoukar I. *MIBE : Méthode d'Ingénierie des Besoins pour l'implantation d'un progiciel de gestion intégré (ERP).* PhD. thesis, Université Paris1 Panthéon Sorbonne, 2005.

ANNEXES

Annexe 1. Description du service $S_{Effectuer\ l'approvisionnement\ en\ produits}$

```
<?xml version="1.0" encoding="UTF-8"?>

<tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
      xmlns:tree='http://ApprovisionnementProduit/Tree'
      xsi:schemaLocation='http://ApprovisionnementProduit/Tree Tree.xsd'>

<tree:And id="and-01">
    <tree:Xor id="xor-01">
         <tree:Leaf id="Sac1">
        </tree:Leaf>
        <tree:And id="and-02">
          <tree:Or id="or-01">
                 <tree:LXor id="lxor-01">
                   <tree:Leaf id="Sab1">
                      <tree:QoS>
                       <tree:Quality>Performance</tree:Quality>
                        <tree:Degree>+</tree:Degree>
                      </tree:QoS>
                   </tree:Leaf>
                    <tree:Leaf id="Sab2">
                      <tree:QoS>
                       <tree:Quality>Performance</tree:Quality>
                         <tree:Degree>++</tree:Degree>
                       </tree:QoS>
                   </tree:Leaf>
                </tree:LXor>
                <tree:Leaf id="Sab3">
                     <tree:QoS>
                      <tree:Quality>Performance</tree:Quality>
                      <tree:Degree>-</tree:Degree>
                    </tree:QoS>
               </tree:Leaf>
           </tree:Or>
            <tree:Leaf id="Sbc1">
            </tree:Leaf>
        </tree:And>
    </tree:Xor>
    <tree:Star id="star-01">
         <tree:Or id="or-02">
              <tree:Leaf id="Scc1">
              </tree:Leaf>
              <tree:Leaf id="Scc2">
              </tree:Leaf>
              <tree:Leaf id="Scc3">
                     <tree:QoS>
                      <tree:Quality>Fiabilité</tree:Quality>
                       <tree:Degree>+</tree:Degree>
                     </tree:QoS>
              </tree:Leaf>
               <tree:LXor id="lxor-02">
                   <tree:Leaf id="Scc4">
                     <tree:QoS>
                      <tree:Quality>Fiabilité</tree:Quality>
```

```
                        <tree:Degree>-</tree:Degree>
                      </tree:QoS>
                    </tree:Leaf>
                     <tree:Leaf id="Scc5">
                       <tree:QoS>
                       <tree:Quality>Fiabilité</tree:Quality>
                         <tree:Degree>+</tree:Degree>
                       </tree:QoS>
                    </tree:Leaf>
                    <tree:Leaf id="Scc6">
                     <tree:QoS>
                       <tree:Quality>Fiabilité</tree:Quality>
                         <tree:Degree>++</tree:Degree>
                      </tree:QoS>
                    </tree:Leaf>
                </tree:LXor>
          </tree:Or>
    </tree:Star>
    <tree:Leaf id="Scd1">
        <tree:QoS>
             <tree:Quality>Fiabilité</tree:Quality>
             <tree:Degree>++</tree:Degree>
        </tree:QoS>
    </tree:Leaf>
</tree:And>
</tree:Services>
```

Annexe 2. La spécification du service $S_{\text{Réaliser analyse plasma}}$

```
<?xml version="1.0" encoding="UTF-8"?>

<tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
      xmlns:tree='http://DiagnosticaStago/Tree'
      xsi:schemaLocation='http://DiagnosticaStago/Tree Tree.xsd'>

<tree:And id="and-01">
  <tree:LXor id="lxor-01">
      <tree:Leaf id= "S Authentifier l'opérateur par mot de passe">
          <tree:QoS>
               <tree:Quality>Confidentialité</tree:Quality>
               <tree:Degree>+</tree:Degree>
          </tree:QoS>
          <tree:QoS>
               <tree:Quality>Temps</tree:Quality>
               <tree:Degree>+</tree:Degree>
          </tree:QoS>
       </tree:Leaf>
       <tree:Leaf id="S Authentifier l'opérateur par clé">
           <tree:QoS>
                <tree:Quality>Confidentialité </tree:Quality>
                <tree:Degree>++</tree:Degree>
           </tree:QoS>
           <tree:QoS>
                <tree:Quality>Temps</tree:Quality>
                <tree:Degree>-</tree:Degree>
           </tree:QoS>
      </tree:Leaf>
  </tree:LXor>

 <tree:Star id="star-01">
```

```
          <tree:Leaf id="SCharger les tubes par mode urgent">
        </tree:Leaf>
 </tree:Star>

   <tree:Or id="or-01">
                  <tree:Leaf id="SIdentifier tubes automatiquement">
                     <tree:QoS>
                        <tree:Quality>précision</tree:Quality>
                        <tree:Degree>+</tree:Degree>
                     </tree:QoS>
                     <tree:QoS>
                        <tree:Quality>temps</tree:Quality>
                        <tree:Degree>+</tree:Degree>
                     </tree:QoS>
                  </tree:Leaf>
                  <tree:Leaf id="SIdentifier tubes par code à barre">
                     <tree:QoS>
                        <tree:Quality>précision</tree:Quality>
                         <tree:Degree>+</tree:Degree>
                     </tree:QoS>
                     <tree:QoS>
                        <tree:Quality>temps</tree:Quality>
                        <tree:Degree>-</tree:Degree>
                     </tree:QoS>
                  </tree:Leaf>
                  <tree:Leaf id="SIdentifier tubes manuellement">
                     <tree:QoS>
                        <tree:Quality>précision</tree:Quality>
                         <tree:Degree>-</tree:Degree>
                     </tree:QoS>
                     <tree:QoS>
                        <tree:Quality>temps</tree:Quality>
                        <tree:Degree>-</tree:Degree>
                     </tree:QoS>
                  </tree:Leaf>
    </tree:Or>


<tree:Xor id="xor-01">

    <tree:Leaf id="SRelancer dernière analyse">
      <tree:QoS>
        <tree:Quality>précision</tree:Quality>
        <tree:Degree>--</tree:Degree>
      </tree:QoS>
      <tree:QoS>
        <tree:Quality>temps</tree:Quality>
        <tree:Degree>--</tree:Degree>
      </tree:QoS>
    </tree:Leaf>

<tree:And id="and-02">

    <tree:LXor id="lxor-02">
                <tree:Leaf id="SRécupérer plasma par centrifugation">
                     <tree:QoS>
                        <tree:Quality>précision</tree:Quality>
                        <tree:Degree>-</tree:Degree>
                     </tree:QoS>
                      <tree:QoS>
```

```
                    <tree:Quality>temps</tree:Quality>
                      <tree:Degree>+</tree:Degree>
                    </tree:QoS>
                </tree:Leaf>
            <tree:Leaf id="SRécupérer plasma par microfiltration">
                    <tree:QoS>
                      <tree:Quality>précision</tree:Quality>
                       <tree:Degree>+</tree:Degree>
                    </tree:QoS>
                     <tree:QoS>
                     <tree:Quality>temps</tree:Quality>
                      <tree:Degree>-</tree:Degree>
                    </tree:QoS>
                </tree:Leaf>
 </tree:LXor>


 <tree:Or id="or-02">
   <tree:Leaf id="STraiter plasma par TQ">
  </tree:Leaf>
  <tree:Leaf id="STraiter plasma par Fib">
  </tree:Leaf>
  <tree:Leaf id="STraiter plasma par VwF">
  </tree:Leaf>
  <tree:Leaf id="STraiter plasma par TCA">
  </tree:Leaf>
   <tree:Leaf id="STraiter plasma par ATIII">
  </tree:Leaf>
  <tree:Leaf id="STraiter plasma par TS">
  </tree:Leaf>
  <tree:Leaf id="STraiter plasma par TP">
  </tree:Leaf>
</tree:Or>

<tree:Or id="or-03">
             <tree:Leaf id="SAnalyser plasma sans incubation">
                    <tree:QoS>
                      <tree:Quality>temps</tree:Quality>
                      <tree:Degree>+</tree:Degree>
                    </tree:QoS>
                </tree:Leaf>
             <tree:Leaf id="SAnalyser plasma avec une incubation">
                    <tree:QoS>
                      <tree:Quality>temps</tree:Quality>
                       <tree:Degree>-</tree:Degree>
                    </tree:QoS>
                </tree:Leaf>
            <tree:Leaf id="SAnalyser plasma avec deux incubations">
                    <tree:QoS>
                      <tree:Quality>temps</tree:Quality>
                       <tree:Degree>--</tree:Degree>
                    </tree:QoS>
                </tree:Leaf>
 </tree:Or>

 <tree:Or id="or-04">
    <tree:Leaf id="SRéaliser l’analyse par mesure chronométrique">
             <tree:QoS>
                      <tree:Quality>temps</tree:Quality>
                       <tree:Degree>+</tree:Degree>
             </tree:QoS>
```

```
                </tree:Leaf>
               <tree:LXor id="lxor-03">
        <tree:Leaf id="SRéaliser l'analyse par mesure immunologique">
                     <tree:QoS>
                        <tree:Quality>temps</tree:Quality>
                        <tree:Degree>-</tree:Degree>
                     </tree:QoS>
                  </tree:Leaf>
      <tree:Leaf id="SRéaliser l'analyse par mesure colorimétrique">
                     <tree:QoS>
                         <tree:Quality>temps</tree:Quality>
                         <tree:Degree>-</tree:Degree>
                     </tree:QoS>
                  </tree:Leaf>
            </tree:LXor>
    </tree:Or>


</tree:And>
</tree:Xor>


   <tree:Or id="or-05">
     <tree:Leaf id="SArrêter par validation résultat">
      <tree:QoS>
         <tree:Quality>précision</tree:Quality>
         <tree:Degree>+</tree:Degree>
      </tree:QoS>
      <tree:QoS>
         <tree:Quality>temps</tree:Quality>
         <tree:Degree>-</tree:Degree>
      </tree:QoS>
     </tree:Leaf>
     <tree:Leaf id="SArrêter par complétude">
      <tree:QoS>
        <tree:Quality>précision</tree:Quality>
        <tree:Degree>-</tree:Degree>
   </tree:QoS>
   <tree:QoS>
      <tree:Quality>temps</tree:Quality>
      <tree:Degree>+</tree:Degree>
   </tree:QoS>
     </tree:Leaf>
   </tree:Or>

</tree:And>
</tree:Services>
```

Annexe 3. Description du service $S_{\text{Réserver un billet de train}}$

```
<?xml version="1.0" encoding="UTF-8"?>

<tree:Services xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
      xmlns:tree='http://RéserverBilletDeTrain/Tree'
      xsi:schemaLocation='http://RéserverBilletDeTrain/Tree Tree.xsd'>

<tree:And id="and-01">
    <tree:LXor id="lxor-01">
         <tree:Leaf id= "S Réserver un billet de train par internet ">
               <tree:QoS>
                        <tree:Quality>commodité</tree:Quality>
```

```
                        <tree:Degree>+</tree:Degree>
                </tree:QoS>

            </tree:Leaf>
              <tree:Leaf id= "S Réserver un billet de train en agence">
                 <tree:QoS>
                          <tree:Quality>commodité</tree:Quality>
                           <tree:Degree>-</tree:Degree>
                 </tree:QoS>
            </tree:Leaf>
    </tree:LXor>
    <tree:Star id="star-01">
          <tree:Leaf id= "S Réserver un billet de train réduit">
                 <tree:QoS>
                          <tree:Quality>cout</tree:Quality>
                           <tree:Degree>+</tree:Degree>
                 </tree:QoS>
            </tree:Leaf>
    </tree:Star>

    <tree:Xor id="xor-01">
          <tree:Leaf id= "S Annuler un billet de train">
          </tree:Leaf>
          <tree:And id="and-02">

              <tree:LXor id=”lxor-02">
                     <tree:Leaf id= "S Payer billet par carte bancaire">
                       <tree:QoS>
                           <tree:Quality>confidentialité</tree:Quality>
                            <tree:Degree>+</tree:Degree>
                       </tree:QoS>
                        </tree:Leaf>
                      <tree:Leaf id= "S Payer billet par chèque">
                       <tree:QoS>
                           <tree:Quality>confidentialité</tree:Quality>
                            <tree:Degree>-</tree:Degree>
                       </tree:QoS>
                     </tree:Leaf>
                        <tree:Leaf id= "S Payer billet par espèce">
                        <tree:QoS>
                           <tree:Quality>confidentialité</tree:Quality>
                            <tree:Degree>-</tree:Degree>
                       </tree:QoS>

                     </tree:Leaf>
              </tree:LXor>
              <tree:Or id="or-01">
                  <tree:Leaf id= "S Présenter une carte de réduction">
                     <tree:QoS>
                           <tree:Quality>cout</tree:Quality>
                            <tree:Degree>+</tree:Degree>
                     </tree:QoS>
                   </tree:Leaf>
                  <tree:Leaf id="S Présenter billet de train">
                    </tree:Leaf>
              </tree:Or>
        </tree:And>
</tree:And>
</tree:Services>
```

Annexe 4. Description du service intentionnel (XSD)

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3      targetNamespace="http://xml.netbeans.org/schema/Tree"
4      xmlns:tns="http://xml.netbeans.org/schema/Tree"
5   elementFormDefault="qualified" >
6
7   <xsd:element name="Services">
8     <xsd:complexType>
9        <xsd:group ref="tns:ServicesChild" minOccurs="1" maxOccurs="1"/>
10    </xsd:complexType>
11  </xsd:element>
12
13  <xsd:element name="And">
14    <xsd:complexType>
15      <xsd:sequence>
16       <xsd:group ref="tns:Child" minOccurs="2" maxOccurs="unbounded"/>
17      </xsd:sequence>
18        <xsd:attribute name="id" use="required" type="xsd:string" />
19    </xsd:complexType>
20  </xsd:element>
21
22  <xsd:element name="Xor">
23    <xsd:complexType>
24      <xsd:sequence>
25       <xsd:group ref="tns:Child" minOccurs="2" maxOccurs="unbounded"/>
26      </xsd:sequence>
27        <xsd:attribute name="id" use="required" type="xsd:string" />
28    </xsd:complexType>
29   </xsd:element>
30
31   <xsd:element name="Or">
32    <xsd:complexType>
33       <xsd:sequence>
34      <xsd:group ref="tns:Leaves" minOccurs="2" maxOccurs="unbounded"/>
35       </xsd:sequence>
36       <xsd:attribute name="id" use="required" type="xsd:string" />
37    </xsd:complexType>
38    </xsd:element>
39
40   <xsd:element name="LXor">
41     <xsd:complexType>
42        <xsd:sequence>
43  <xsd:group ref="tns:LXorLeaves" minOccurs="2" maxOccurs="unbounded"/>
44        </xsd:sequence>
45     <xsd:attribute name="id" use="required" type="xsd:string" />
46     </xsd:complexType>
47   </xsd:element>
48
49   <xsd:element name="Star">
50     <xsd:complexType>
51       <xsd:group ref="tns:ServicesChild" minOccurs="1" maxOccurs="1"/>
52       <xsd:attribute name="id" use="required" type="xsd:string" />
53     </xsd:complexType>
54   </xsd:element>
55
56  <xsd:group name="Child">
57          <xsd:choice>
58                  <xsd:element ref="tns:And" />
```

```
59                  <xsd:element ref="tns:Xor" />
60                  <xsd:element ref="tns:Or" />
61                  <xsd:element ref="tns:Leaf" />
62                  <xsd:element ref="tns:Star" />
63                  <xsd:element ref="tns:LXor" />
64           </xsd:choice>
65       </xsd:group>
66
67       <xsd:group name="ServicesChild">
68           <xsd:choice>
69                  <xsd:element ref="tns:And" />
70                  <xsd:element ref="tns:Xor" />
71                  <xsd:element ref="tns:Or" />
72                  <xsd:element ref="tns:Leaf" />
73                  <xsd:element ref="tns:LXor" />
74           </xsd:choice>
75       </xsd:group>
76
77       <xsd:group name="Leaves">
78           <xsd:choice>
79                  <xsd:element ref="tns:Leaf" />
80                  <xsd:element ref="tns:LXor" />
81                  <xsd:element ref="tns:Star" />
82           </xsd:choice>
83       </xsd:group>
84
85       <xsd:group name="LXorLeaves">
86           <xsd:choice>
87                  <xsd:element ref="tns:Leaf" />
88                  <xsd:element ref="tns:Star" />
89           </xsd:choice>
90       </xsd:group>
91
92       <xsd:element name="Leaf">
93           <xsd:complexType>
94               <xsd:sequence>
95     <xsd:element ref="tns:QoS" minOccurs="0" maxOccurs="unbounded"/>
96               </xsd:sequence>
97            <xsd:attribute name="id" use="required" type="xsd:string" />
98           </xsd:complexType>
99       </xsd:element>
100
101
102      <xsd:element name="QoS">
103          <xsd:complexType>
104              <xsd:sequence>
105       <xsd:element ref="tns:Quality" minOccurs="1" maxOccurs="1"/>
106       <xsd:element ref="tns:Degree" minOccurs="1" maxOccurs="1"/>
107              </xsd:sequence>
108          </xsd:complexType>
109      </xsd:element>
110
111      <xsd:element name="Quality" type="xsd:string"/>
112      <xsd:element name="Degree" type="xsd:string"/>
113
114  </xsd:schema>
```