Approche orientée modèles pour la vérification et l'évaluation de performances de l'interopérabilité et l'interaction des services

Wafaa Ait-Cheik-Bihi

Nř d'ordre : 182 Année : 2012

THÈSE

Pour l'obtention du grade de

DOCTEUR DE L'UNIVERSITE DE TECHNOLOGIE DE BELFORT-MONTBELIARD

(Spécialité Informatique)

ÉCOLE DOCTORALE SCIENCES POUR L'INGENIEUR & MICROTECHNIQUES

Présentée par

Wafaa AIT-CHEIK-BIHI

Approche orientée modèles pour la vérification et l'évaluation des performances de l'interopérabilité et l'interaction des services

Soutenue le 21 Juin 2012

Devant le jury

Présidente :	
Pr. Dominique Faudot	Université de Bourgogne, Dijon
Rapporteurs :	
Pr. Pascal Lorenz	Université de Haute Alsace (UHA), Mulhouse
Pr. Jean Louis Boimond	Université d'Angers (ISTIA), Angers
Directeurs :	
Dr. Maxime Wack	Université de Technologie de Belfort-Montbèliard, Belfort
Dr. Ahmed Nait-Sidi-Moh	Université de Picardie Jules Verne (INSSET), Saint-Quentin
Examinateurs :	
Pr. Alexandre Caminada	Université de Technologie de Belfort-Montbèliard, Belfort
Dr. Mohamed Bakhouya	Aalto University School of Engineering, Finland
Dr. Jaafar Gaber	Université de Technologie de Belfort-Montbèliard, Belfort

Remerciement

Ces travaux de thèse se sont déroulés au sein du laboratoire Systèmes et Transport (SET) à l'Université de Technologie de Belfort-Montbèliard.

Je tenais en premier lieu à remercier vivement Maxime Wack, mon directeur de thèse, et Ahmed Nait-Sidi-Moh, mon co-encadrant, pour leurs collaborations, leurs disponibilités et leurs précieux conseils qui m'ont permis d'enrichir mon travail, je les remercie également pour leur soutien tout au long du déroulement de ma thèse. Ils m'ont également offert l'opportunité de participer à plusieurs projets de recherche Européens : ASSET et TELEFOT. Ceci a été très enrichissant et m'a permis de rencontrer de nombreux partenaires issus des différents pays de l'Union Européenne.

J'adresse mes profondes reconnaissances à Monsieur Jean-Louis Boimond Professeur à l'université d'Angers et Monsieur Pascal Lorenz Professeur à l'université de haute Alsace d'avoir accepté d'être les rapporteurs des travaux présentés, je les remercie pour le temps qu'ils ont consacré à cette tâche, l'intérêt qu'ils ont porté à mon travail, les remarques enrichissantes qu'ils ont formulées ainsi que d'avoir accepté de juger mon travail en participant au jury.

Toute ma reconnaissance va à Madame Dominique Faudot Professeur à l'université de Bourgogne, aux Messieurs Alexandre Caminada Professeur à l'université de technologie de Belfort-Montbéliard, Jaafar Gaber Maître de conférences à l'université de technologie de Belfort-Montbéliard, et Mohamed Bakhouya Chercheur à l'école d'ingénieur de l'université d'Aalto en Finlande pour l'intérêt qu'ils ont bien voulu porter à ce travail en acceptant de l'examiner.

Je tiens aussi à adresser ma gratitude à toutes les personnes qui m'ont aidé pendant la préparation de cette thèse, je pense notamment à tous les membres de l'équipe (Mohamed, Jaafar) qui m'ont permis de progresser dans mon travail de recherche, des personnes ayant collaboré de prés ou de loin à ce travail : mes collègues, mes amis, et le personnel du département

informatique.

Enfin, je ne saurais terminer cette liste, sans adresser un remerciement particulier et chaleureux à toute ma famille, plus particulièrement : ma mère, mon frère, mes soeurs et mon grand-père pour leur soutien moral et leurs conseils malgré la distance qui nous sépare. Et je fini par exprimer toute mon affection à mon époux Charaf-Eddine, qui m'a apporté son soutien lors de ces années de thèse.

Table des matières

Introduction générale

1 Contexte du travail

Cadre du travail : L'interopérabilité et la collaboration de services Web pour l'amélioration de la sécurité routière.

Cadre précis : Mise en oeuvre d'une approche orientée modèles pour la vérification formelle et l'évaluation de performances des processus d'interopérabilité et d'interaction dans la composition de services Web.

2 Problématique traitée

Actuellement, les services Web sont très utilisés notamment par les entreprises pour rendre accessible leurs métiers ou leurs données *via* le Web. L'émergence de ces services a permis aux applications web d'être vues comme un ensemble de services métiers bien structurés et correctement décrits, plutôt qu'un ensemble d'objets et de méthodes. En effet, cela facilite les échanges entre les applications d'une même organisation, mais permet aussi l'ouverture vers les applications des autres organisations d'une façon distribuée ; ce qui permet d'offrir des avantages primordiaux en termes d'interopérabilité et de maintenance de telles applications.

La composition de services sert à définir et à faciliter le processus d'intégration et de collaboration entre plusieurs services distribués. Cette technologie émergente présente beaucoup d'intérêts, tant dans le domaine de la recherche que dans le domaine industriel.

L'une des plus importantes et complexes tâches de l'architecture SOA (l'acronyme en anglais de *Service Oriented Architecture*) est de créer des services à valeur ajoutée ; c'est-à-dire

des services composés qui sont le résultat de la composition de plusieurs services existants. La complexité de cette tâche provient de l'hétérogénéité des données des différents services à composer et de leurs formats (p. ex. integer, float, type complexe). Elle est due également à la nature variable et dynamique de l'environnement des services Web. En effet, le nombre de fournisseurs de services augmente constamment et, de nouveaux services deviennent, de plus en plus disponibles. Idéalement, les services à composer devraient être capables de s'adapter de manière transparente aux changements de l'environnement et aux exigences des consommateurs avec un minimum d'intervention humaine. Par conséquent, la difficulté majeure réside dans la composition dynamique de services, en particulier lorsqu'une tâche, indispensable pour un scénario, ne peut pas être réalisée par le service existant.

Des langages de composition, tels que BPEL ou WSCI proposés dans la littérature, sont textuels et sont principalement utilisés pour la composition de services dans un environnement statique, c'est-à-dire la composition est effectuée à partir d'un scénario réalisé au préalable où tous les services participants sont connus à l'avance lors de la conception du scénario. A cela, il faut ajouter que la spécification et la vérification formelles de la composition de services ne sont pas prisent en compte par ces langages. Il est donc nécessaire de développer des méthodes et des modèles formels pour l'étude de faisabilité, l'étude de performances ainsi que la validation des processus de composition de services. Cette étape facilitera la maîtrise totale du processus de développement de bout en bout. Le travail présenté dans ce mémoire aborde cette problématique en proposant une approche orientée modèles pour la spécification, la vérification formelle et l'évaluation de performances des workflows patterns qui représentent des processus de composition de services.

3 Contributions de la thèse

Dans ce mémoire, une approche orientée modèles est proposée pour spécifier, valider et mettre en oeuvre la composition de services Web. Cette approche est basée sur deux outils formels, à savoir les Réseaux de Petri (RdP) et l'algèbre (max,+) ; ces outils sont utilisés pour modéliser, évaluer les performances et vérifier le processus de la composition de services de façon automatique et dynamique. La composition est considérée automatique du fait que le scénario de composition n'est pas identifié lors de l'étape de développement mais plutôt élaboré à la volée au moment de l'exécution. De même, elle est dynamique puisque les services sont identifiés au moment de l'exécution. Pour cela, nous utilisons les workflows patterns dans le domaine de sécurité routière dans les transport, où chaque pattern est représenté par un modèle RdP puis,

traduit en une équation mathématique dans l'algèbre (max,+). Notons que les workflow pattern permettent la modélisation et la gestion automatique de l'ensemble des tâches à accomplir dans un processus métier ainsi que les différents acteurs impliqués dans la réalisation de ce processus.

Une plateforme appelée - TransportML - pour la collaboration et l'interopérabilité de services à base de positionnement, (LBS : l'acronyme en anglais de *Location-Based Service*), est implémentée, et ses fonctionnalités sont utilisées pour répondre au mieux aux exigences de l'utilisateur en se basant sur la composition de services. Les résultats des simulations obtenus à partir des modèles formels décrivant et évaluant des scénarios de composition de services, sont comparés à, d'une part à ceux obtenu à partir de l'exécution, sur machine, de la plateforme TransportML, et d'autre part, à ceux issus des expérimentations réelles sur le terrain.

Le travail présenté dans cette thèse s'inscrit dans le cadre des deux projets Européens ASSET et TeleFOT. Le projet ASSET (*Advanced Safety and Driver Support for Essential Road Transport*, www.project-asset.com, 07/2008 - 12/2011), est un projet Européen du programme FP7 dont l'objectif principal est d'intégrer les différentes technologies (systèmes de positionnements, technologies de communication, service Web, les réseaux mobiles, ...) pour l'amélioration de la sécurité routière de façon holistique. Quant au projet Européen TeleFOT du programme FP7 (Field Operational Tests of Aftermarket and Nomadic Devices in Vehicles, http ://www.telefot.eu/, 06/2008 - 12/2012), il se focalise sur l'étude de l'impact des périphériques mobiles (Smart phones, GPS, etc.) embarqués dans les véhicules pour la délivrance des informations pertinentes, l'assistance aux conducteurs en cas d'urgence (e-Call), et la sensibilisation des conducteurs aux fonctionnalités et aux potentiels de ces périphériques pour la sécurité routière [ACBCB+11].

4 Organisation du mémoire

4.1 Présentation générale

Le présent document est constitué de six chapitres. Le premier chapitre présente un rappel sur l'architecture orientée service (SOA), les services Web, l'approche MDA (*Model Driven Architecture*) et les LBS. Le deuxième chapitre donne un état de l'art des approches dirigées par les modèles pour la composition de services. Le troisième chapitre est consacré à la présentation ainsi qu'à la modélisation formelles des workflows patterns utilisés dans la composition de services en se basant sur les RdP et l'algèbre (max,+). Le quatrième chapitre présente la plateforme TransportML. Le cinquième chapitre porte sur l'illustration et la validation de l'approche formelle proposée, à travers des scénarios de composition de services, par des simulations et

des expérimentations et tests sur le terrain. Pour finir, les perspectives et les conclusions de ces travaux sont présentées dans le dernier chapitre.

4.2 Plan global

- Chapitre1 : Approches de la composition de services
- Chapitre2 : Approches orientées modèles pour la composition de services
- Chapitre3 : Spécification et modélisation formelles de la composition de services
- Chapitre4 : TransportML pour la collaboration des LBS
- Chapitre5 : Expérimentations et résultats
- Chapitre6 : Conclusions et perspectives

4.3 Contenu des chapitres

Chapitre1

Dans ce chapitre, l'architecture orientée service (SOA) est introduite ainsi que la description des services Web, nous soulignons brièvement les différents composants participants à la mise en oeuvre d'une telle architecture. Nous proposons également quelques définitions de l'approche MDA en explicitant ses différents composants. Une description détaillée est présentée au sujet des LBS. Nous abordons enfin, le domaine de la composition de services, en décrivant dans un premier temps les approches de la composition de services et en citant ensuite les différents langages utilisés pour chaque approche.

Chapitre2

Dans ce chapitre, nous étudions des approches orientées modèles pour la composition de services. Nous proposons ensuite, une classification de ces approches selon leurs fonctions et leur mode de fonctionnement. Puis, nous comparons les différentes approches citées dans cette classification. Nous terminons le chapitre par une synthèse en positionnant notre travail par rapport à des travaux de recherche développés dans le domaine.

Chapitre3

Ce chapitre décrit les workflows patterns les plus utilisés pour la composition de services Web. Les comportements de ces workflows patterns sont présentés par la suite par des modèles RdP pour l'étude de faisabilité et la vérification formelle de certaines propriétés qualitatives.

Ces modèles sont traduits, ensuite, par des équations dans l'algèbre (max,+) pour l'étude de performance des métriques quantitatives. Plusieurs critères de la qualité de service (QoS) pour la composition automatique de services Web sont pris en considération lors de la phase de validation.

Chapitre4

Ce chapitre présente l'architecture de la plateforme TransportML que nous avons développé dans ce travail. Ses différents composants et leurs interactions sont décrits en détail. Le protocole de communication entre ces composants est également défini.

Chapitre5

Dans ce chapitre, nous avons modélisé, analysé et évalué un des scénarios, de composition de services, considérés dans les projets Européens ASSET et TeleFOT. Il s'agit de faire collaborer plusieurs services à des fins de sécurité routière. Des tests et des expérimentations sur le terrain ont été réalisés ainsi qu'une analyse formelle par simulation est rapportée dans ce chapitre. Enfin, une étude comparative des résultats obtenus par les deux approches théorique et pratique est établie.

Chapitre6

Ce chapitre présente les conclusions de notre travail et les travaux futurs d'amélioration et d'extension à mener dans ce domaine.

Chapitre I

Approches de la composition de services

Dans ce chapitre, nous présentons les différentes terminologies et technologies de composition de services utilisées dans ce travail. Nous allons voir également les différentes approches et modèles présentés dans la littérature pour la composition de services. Dans la section 1, nous introduisons l'architecture SOA en exposant ses principes et en détaillant les différentes parties mettant en oeuvre cette architecture. La section 2 traitera quant à elle des architectures orientées ingénierie et orientées modèles. Pour finir, la section 3 regroupe les approches de composition de services en détaillant les différents types de composition.

1 Architecture Orientée Service

L'architecture orientée service SOA [1] (Service Oriented Architecture) est une architecture qui a vu le jour au début de l'année 2000 [Erl07], pour répondre aux besoins des entreprises en termes de décentralisation et de répartition des applications logicielles. A l'origine, celle ci a été essentiellement utilisée pour résoudre les problématiques d'interopérabilité entre les différentes technologies informatiques distribuées utilisées en entreprise. C'est une solution très efficace en termes de réutilisabilité, d'interopérabilité et de réduction de couplage entre les différentes entités implémentant leurs propres systèmes d'information. SOA est un modèle d'interaction applicative qui permet de décomposer une application en un ensemble de composants basiques (c.-à-d. services), de décrire le schéma d'interaction entre ces composants en utilisant un format d'échange bien défini, le plus souvent XML, et des couplages externes par l'intermédiaire d'une couche d'interopérabilité des interfaces, le plus souvent un service Web.

L'architecture SOA a été popularisée avec l'apparition des standards comme les services Web dans l'e-commerce, B2B (Business to Business) ou B2C (Business to Client). L'élément clé de la mise en oeuvre de cette architecture est de rendre accessibles, *via* un réseau (par exemple

1. Les principes de SOA : http ://www.soaprinciples.com/

Internet), les services Web (§1.1) afin que les utilisateurs puissent accéder à ces services et les réutiliser dans leurs applications. Ces services sont exposés au travers de contrats respectant les mêmes règles de standardisation. Ces contrats ne contiennent que les informations essentielles permettant l'invocation des services. Seules ces informations doivent être publiées. Un des avantages principaux de cette architecture est qu'elle définit les préceptes de la découverte dynamique des services, leur sélection et leur intégration. En effet, SOA encourage la réutilisation des services déployés à grande échelle dans les systèmes basés sur cette architecture, et par conséquent le développement logiciel devient plus économique et plus efficace.

1.1 Service Web

Un Service Web [Cha02] est un composant logiciel autonome qui reçoit des requêtes et qui renvoie des réponses au travers une interface standard bien définie. Les services Web sont indépendants de toute plateforme d'exécution et de tout langage de programmation. Plus précisément, un service Web est un composant logiciel autonome et unique (définit par une seule instance de service) qui permet de décrire un ensemble d'opérations accessibles *via* le Web. Il se base sur le standard XML pour décrire les appels de fonctions distantes et les données échangées en utilisant le protocole HTTP comme moyen de communication. Ce protocole est basé sur le principe de requêtes/réponses décrites avec des messages XML.

L'émergence des services Web a permis aux applications d'être vues comme un ensemble de services métiers bien structurés et correctement décrits, plutôt qu'un ensemble d'objets et de méthodes. Cela facilite, non seulement, les échanges entre les applications d'une même organisation mais aussi, les échanges entre les applications des organisations distantes et distribuées. La granularité de cette approche (c.-à-d. le découpage d'une application en services) a un avantage très important en termes de maintenance et d'interopérabilité d'applications. En effet, elle permet de modifier plus facilement un service ou de le remplacer par un autre, réduisant ainsi la complexité d'une application. En d'autres termes, le développeur peut se focaliser sur un service indépendamment du reste de l'application.

Le fonctionnement d'un service Web repose sur un modèle en couches dont les trois couches principales sont représentées dans la figure I.1.

La couche *découverte* a pour objectif de rechercher et de localiser un service Web. Ceci est possible grâce au protocole standard de découverte UDDI (*Universal Description Discovery and Integration*). La deuxième couche - *Description* - permet de décrire les interfaces des services Web (c.-à-d. les paramètres des fonctions, les types de données, etc.). Les services Web offrent un moyen de décrire leurs interfaces d'une manière suffisamment détaillée pour permettre à un

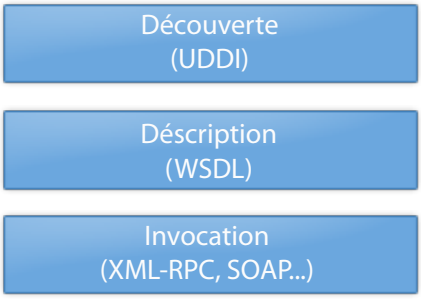


Figure I.1 – Service Web - Modèle en couches

utilisateur de créer une application cliente capable de communiquer avec eux. La description d'un service est généralement fournie par un document XML nommé WSDL (*Web Service Description Language*) qui précise les méthodes pouvant être invoquées, leurs signatures et les points d'accès du service (URL, port, etc.). La dernière couche, dite couche d'*invocation*, décrit alors la structure des messages échangés. En effet, les services Web proposent aux utilisateurs des fonctionnalités pratiques grâce à un protocole standard basé sur XML ; dans la plupart des cas, le protocole utilisé est SOAP (*Simple Object Access Protocol*) [GHM+11] ou encore XML-RPC (*XML Remote Procedure Call*). Ces technologies seront décrites dans la section suivante.

1.2 Architecture standard des services Web

Les services Web fournissent un moyen standard d'échange de données entre différentes applications logicielles fonctionnant sur une variété de plateformes et/ou de *frameworks*. L'architecture des services Web est une architecture qui identifie des éléments globaux nécessaires pour assurer l'interopérabilité entre les services. Ces éléments correspondent aux notions d'annuaire, de bus, de contrat et de protocole de communication. Cette architecture est illustrée dans la Figure I.2.

Des informations sur un service sont stockées dans un endroit bien défini, UDDI, et le client récupère, à travers cet annuaire, les informations relatives au service avec qui il souhaite communiquer. Cette communication se base sur un format commun (SOAP), en s'appuyant sur l'interface de description du service (WSDL). Une courte description de chacun de ces standards est exposée ci-après.

- SOAP [2] est un protocole de RPC (Remote Procedure Call) permettant d'invoquer des

2. SOAP, http ://www.w3.org/TR/soap12-part1/

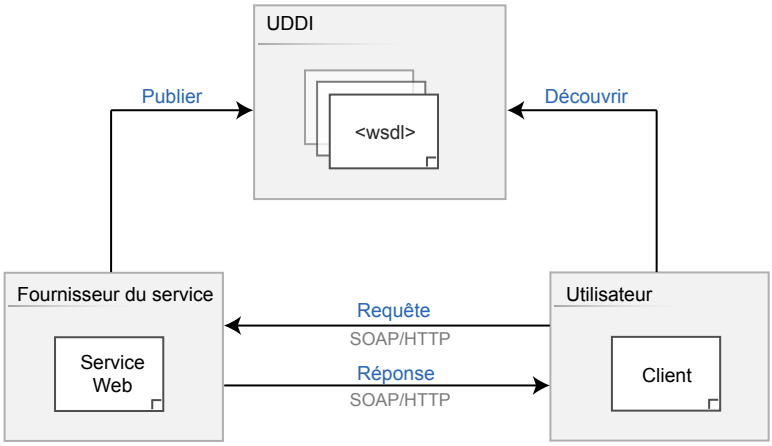


Figure I.2 – Une Architecture de service Web basique

services distants. Il a été définit à l'origine par Microsoft, puis standardisé par W3C. Il permet de définir les mécanismes d'échanges d'information entre des clients et des fournisseurs de service Web. Le principe de fonctionnement de SOAP est comparable à celui de DCOM (Distributed Component Object Model) ou de CORBA (Common Object Request Broker Architecture) mais contrairement à ces deux standards, SOAP s'appuie sur des standards très connus. Il utilise XML pour définir les fonctions disponibles. De plus, il prend en charge divers protocoles de transport, tels que HTTP et SMTP, ainsi que différents formats comme MIME [3] (Multipurpose Internet Mail Extensions). Ces derniers sont très répandus sur de multiples plates-formes, ce qui donne au protocole SOAP une grande portabilité et interopérabilité. SOAP est une spécification non propriétaire et n'est pas lié à un protocole particulier. Il n'est pas non plus lié à un système d'exploitation ni à un langage de programmation. Étant un protocole d'échange d'informations entre diverses machines sur un réseau, il nécessite un format pour transporter les données. De ce fait, un message SOAP est un document XML comportant une enveloppe et un corps.

- UDDI [4] (Universal Description Discovery and Integration) est né d'une collaboration entre IBM [5] (International Business Team) et ARIBA [6], et ensuite définit par OISIS. Il normalise une solution d'annuaires distribués de services Web permettant, d'une façon standard, à la fois de publier et d'explorer les services, soit pour une utilisation sur un réseau public, soit dans l'infrastructure interne d'une organisation. UDDI offre un mécanisme fondé sur des normes pour classer, cataloguer et gérer les services Web de tel sorte qu'ils peuvent être découverts et

3. MIME, Un standard d'Internet qui étend le format émail pour, entre autres, supporter d'autres codages que ASCII.
4. UDDI, http ://uddi.xml.org/
5. http ://www.ibm.com/us/en/
6. ARIBA, http ://www.ariba.com/

utilisés par d'autres applications.

- WSDL[7] (Web Service Description Language) est un langage se basant sur XML qui permet de décrire toutes les informations nécessaires pour invoquer un service Web, à savoir la description du contenu des messages, l'endroit où le service est disponible et le protocole de communication à utiliser pour communiquer avec le service. WSDL utilise la notation XML pour décrire les formats des messages des requêtes/réponses, ce qui le rend indépendant de tout langage de programmation et de toute plateforme (p. ex. système d'exploitation). En résumé, WSDL est un contrat entre un client et un serveur sans dépendance particulière pour une plateforme ou un langage. Ce contrat fait état des spécifications d'interfaces définissant le service Web, en particulier les opérations qu'il réalise et le type de message échangé.

1.3 Services à base de positionnement(LBS)

Un LBS (*Location-Based Service*) est une classe de service Web permettant d'intégrer des informations géographiques dans les applications métiers. Il fournit des informations accessibles depuis des appareils mobiles *via* le réseau mobile en faisant usage de la position géographique de l'appareil. Les LBS permettent d'identifier l'emplacement d'un objet ou d'une personne par rapport à une position géographique donnée. Ils vont bien sûr au delà d'un simple affichage sur une carte. En effet, ils ont pour mission d'envoyer une information riche, utile et en temps réel, aux utilisateurs à travers des services qui utilisent la localisation de l'utilisateur ou son adresse pour traiter ses requêtes et afficher le résultat sur une carte. Nous citons, par exemple, le calcul d'un itinéraire, la recherche des restaurants qui se trouvent à proximité, etc.

Grâce à l'émergence et au développement des technologies mobiles, les systèmes de géo-positionnement et les informations de localisation, les LBS sont de plus en plus répandus et disponibles. Ces systèmes ont un potentiel important pour améliorer, par exemple, les services publics tels que l'intervention des services de secours, des pompiers et des policiers en cas d'un accident sur la route. Certains des facteurs qui contribuent à cette évolution sont le développement d'Internet et des technologies connexes, la compréhension et l'exploitation du potentiel des métiers contribuant à ce développement, l'impulsion des *frameworks* et des technologies de l'e-commerce ainsi que l'impressionnante croissance des réseaux mobiles sans-fil [TVM+03].

Les LBS ont besoin d'une infrastructure spécifique afin de localiser un terminal mobile, de fournir des réponses à une requête donnée, d'avoir accès à ces informations et finalement de représenter cette information sur une carte. La structure des LBS est définit dans la section suivante.

7. WSDL, http ://www.w3.org/TR/wsdl20/

1.3.1 Besoins et exigences des systèmes basés sur les LBS

Un système basé sur les LBSs se décompose en quatre couches basiques, les systèmes et techniques de positionnement, les technologies de communication, le Système d'Information Géographique (SIG) et finalement le processus métier du LBS. Ces couches sont définies dans la figure I.3 et détaillées par la suite.

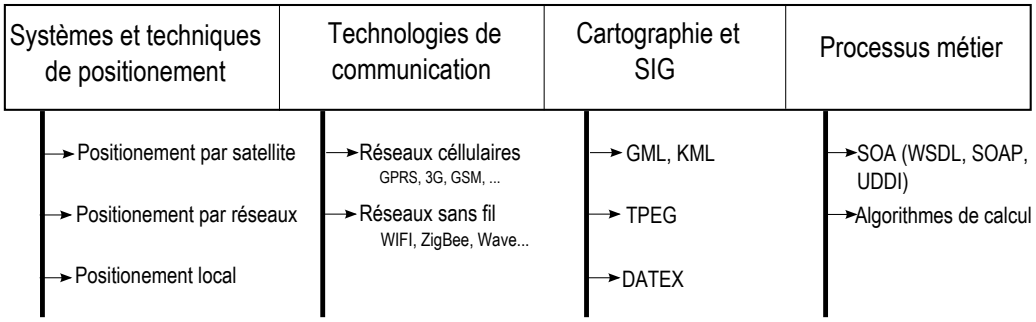


Figure I.3 – Structuration et exigences des systèmes basés sur les LBS

- Systèmes et techniques de positionnement

Différents systèmes et techniques de positionnement varient en fonction de leurs caractéristiques, tels que l'exactitude, la fiabilité et la latence. Ces systèmes peuvent être classés en trois principales catégories : le positionnement par satellite, le positionnement basé sur le réseau et le positionnement local, comme indiqué dans la figure I.3. Le premier type des systèmes de positionnement se base sur les systèmes globaux de navigation par satellite (connus sous le nom GNSS - Global Navigation Satellite System) [WNSMG+09]. Ce type de systèmes permet de déterminer l'emplacement des objets (véhicules, personnes, etc.) en utilisant des signaux radio transmis par les satellites. Parmi ces systèmes, nous citons le système américain NAVSTAR Global Positioning System (GPS) qui est le principal système de positionnement mondial actuel, le système Européen Galileo qui sera entièrement opérationnel en 2013 et le système russe GLONASS. Le deuxième type de positionnement, basé sur les réseaux cellulaires, se réfère à des méthodes de positionnement où les réseaux de télécommunications mobiles sont fortement utilisés pour assurer et fournir les positions des terminaux mobiles. Parmi les approches ou les techniques utilisées pour ce type de positionnement, on trouve la triangulation, l'angle d'arrivée, etc., [RGWNSM07]. Finalement, les systèmes de positionnement locaux opèrent seulement dans une zone restreinte de transmission du signal. Ils couvrent en particulier les LBS dans les environnements intérieurs (c.-à-d. indoors) tels que les grands bâtiments, les centres commerciaux, les tunnels, etc. où le satellite ou le réseau mobile ne sont pas précis ou ne peuvent être

utilisés. Les méthodes de positionnement local comprennent les techniques de positionnement où entre autres WLAN, Bluetooth, RFID peuvent être utilisés. Une des techniques de positionnement utilisé pour ce type de système est A-GPS [RGWNSM07], d'autres approches sont décrites dans [LDBL07].

- Techniques de communication

Les applications basées sur les LBS utilisent des techniques de communication afin de permettre aux utilisateurs un accès facile, *via* des appareils sans fil (p. ex. téléphones mobiles, ordinateur portable, etc.), à des informations pertinentes *via* un réseau. Les technologies GPRS et WAVE sont appropriées pour le déploiement d'applications de géolocalisation. Les deux technologies sont bien adaptées à la mobilité, avec des propriétés différentes. Le système GPRS est une technologie de communication cellulaire, dont l'infrastructure, sur laquelle se base le système GPRS, est présente et bien répandue dans le monde. Ainsi, GPRS offre une portée de communication très élevée avec un coût de déploiement faible. Par contre, il se caractérise par une latence haute et un faible débit. Alors que, WAVE (802.11p) fournit une faible latence et un haut débit sur un rayon pouvant atteindre 1000 mètres. ZigBee est une autre technique de communication qui est extrêmement fiable dans un environnement *Indoor*. Si l'installation de ZigBee est correctement réalisée, cette technologie pourrait être mieux utilisée pour les LBS que la technique de triangulation pour WLAN. ZigBee est très précis et sa mise en oeuvre n'est pas coûteuse. Le développement rapide de ces technologies ainsi que les avantages qu'elle présente incite divers fournisseurs de services à promouvoir la mise en oeuvre des capteurs ZigBee dans leurs appareils. Plus d'informations sur les technologies de communication dans le domaine de transport sont exposés dans le travail [DBG+10], où chaque technique est détaillée avec ses caractéristiques, ses avantages et ses limites. Dans nos travaux de recherche, nous avons utilisé la technologie de communication GSM/GPRS pour réaliser les tests et les expérimentations sur le terrain.

- Cartographie et systèmes d'information géographiques

Les LBSs se basent sur des Systèmes d'Information Géographiques (SIG) pour représenter les données de géolocalisation. Ces systèmes permettent de gérer et d'organiser des informations géographiques localisées, et de disposer les objets dans un système géoréférencé. Ils permettent également de faciliter la superposition des cartes de sources différentes, et d'avoir accès aux informations de tous les objets géographiques situés à une distance donnée d'une route ou de déterminer l'itinéraire le plus court pour se rendre à un endroit précis. Il est important de noter que les LBS se basent principalement sur ces systèmes pour l'utilisation et la représentation

des données géolocalisées. Les données fournies sont utilisées par des algorithmes de calculs, par exemple, pour la représentation d'un itinéraire. Ajoutant à cela que les LBS bénéficient des avantages des SIG. En effet, ces systèmes fonctionnent avec des zones géographiques à grande échelle, par exemple une ville ou un pays. La collecte et la conversion des données utilisées par les SIG produisent d'importantes informations qui peuvent être utilisées par les LBS. En effet, beaucoup de données sont disponibles dans différents systèmes de coordonnées mondiaux (p. ex. WGS84 qui est le système de coordonnées le plus utilisé en France notamment avec le GPS). Un SIG intègre ces données et les convertit en un modèle qui pourrait être représenté sur une carte. Par exemple, en ce qui concerne les applications développées dans le cadre de ce travail, nous utilisons comme système de cartographie la carte de *Google Maps* pour la représentation des données. Quant au système de coordonnées pour les données GPS, nous avons utilisé *WGS84*.

Parmi les langages utilisés par les SIGs, nous citons GML, KML, TPEG, DATEX, comme indiqué dans la figure I.3. GML (*Geography Markup Language*) [Bur06], définit par *Open Geospatial Consortium* (OGC), est un langage qui se base sur l'utilisation de XML pour exprimer des caractéristiques géographiques. C'est un langage de modélisation pour les SIG, ainsi qu'un format d'échange ouvert pour les transactions géographiques sur Internet.

KML [8] (*Keyhole Markup Language*), rendu populaire par *Google*, est un langage complément à GML. Il est plutôt utilisé pour la visualisation des informations géographiques pour les outils de *Google* tels que *Google Earth*.

TPEG [9] (*Transport Protocol Experts Group*) est une famille de spécifications élaborée par EBU (*European Broadcasting Union*) pour la transmission des informations concernant le trafic et le voyage. TPEG est actuellement disponible en deux versions, un format de données binaires et une version basée sur le standard XML appelée tpegML. TPEG a été normalisé par l'ISO pour les messages, les informations sur le trafic, les transports publics et les applications de repérage. Il est déjà implémenté chez certains constructeurs automobiles comme Audi ou BMW.

DATEX a été développée pour l'échange d'informations entre les centres de gestion du trafic, les centres d'information du trafic et les fournisseurs de services. Cette norme a été élaborée par un groupe de travail européen mis en place pour normaliser l'interface entre le contrôle du trafic et les centres d'information. La nouvelle norme DATEX II [10], est devenue une référence pour tous les types d'applications nécessitant un accès à la circulation dynamique et l'information du voyage. Aujourd'hui, DATEX II a établit une position forte comme un standard Européen pour l'échange dynamique de données entre les opérateurs routiers et les fournisseurs de services.

8. KML, http ://www.opengeospatial.org/standards/kml/
9. TPEG, http ://www.tisa.org/technologies/tpeg/
10. DATEX II, http ://www.datex2.eu/

- Processus métier

La dernière couche constitue le coeur de métier du service. Elle définit la principale fonctionnalité des LBS en faisant appel aux technologies définies dans la section 1.1. En effet, un LBS est un service qui comprend et utilise donc tous les éléments définis pour les services Web. Cette couche intègre également les algorithmes de calcul, de fusion et de conversion de données.

1.3.2 Domaines d'application des LBS

Actuellement, les LBS sont utilisés dans une variété de domaines, tels que la santé, la sécurité routière, les assurances. Nous avons regroupé les domaines d'applications des LBS en quatre domaines principaux :

- Les services d'information et de navigation qui fournissent directement les données aux utilisateurs finaux *via* des appareils mobiles en s'appuyant sur les réseaux mobiles. Ces services, sensibles essentiellement à la localisation, font référence à l'information distribuée et basée sur l'emplacement du périphérique, la spécification temporelle et le comportement des utilisateurs. Parmi les exemples d'application de ces services : la localisation de la destination et les critères d'un itinéraire optimisé, les informations du trafic en temps réel, les notifications des lieux d'intérêt.

- Les services d'urgence sont l'une des applications évidente des LBS qui fournissent la possibilité de localiser un individu ignorant son emplacement exact ou qui est dans l'incapacité de le révéler en raison d'une situation d'urgence (p. ex. accident, panne). Grâce à cette information transférée automatiquement aux services des urgences, l'assistance pourrait être fournie avec rapidité et efficacité.

- Les services de tracking se basent généralement sur un système de localisation automatique composé d'un récepteur GPS et d'un module GSM/GPRS. Ce système communique en temps réel et régulièrement les positions du mobile. Ces services peuvent être appliqués, par exemple, aux suivis des colis postaux permettant à ce que leurs positions soient connues à tout moment. Une application similaire est de permettre aux entreprises de localiser leurs personnels sur le terrain afin qu'elles soient en mesure d'alerter l'employé le plus proche d'une intervention. Une autre application des services de tracking dans le domaine de production où l'objectif est de fournir un suivi précis des produits à travers la chaîne d'approvisionnement, ce qui offre de nouvelles possibilités à leur meilleure gestion.

- Les services liés au réseau mobile où la localisation est réalisée en utilisant un récepteur GPS embarqué dans un téléphone mobile ou en utilisant les réseaux de communication. La

connaissance de la position de l'utilisateur peut être améliorée grâce aux services de communication, dans le cas où la couverture GPS n'est pas bonne.

2 Ingénierie et Architecture Orientés Modèles

L'ingénierie logicielle s'oriente actuellement vers l'ingénierie des modèles, après l'approche objet, où chaque artefact logiciel est considéré comme un modèle. L'ingénierie dirigée par les modèles MDE (*Model Driven Engineering*) présente une approche de développement devenue très populaire dans l'ingénierie logicielle, qui se concentre sur la création et l'exploitation de modèles abstraits. En d'autres termes, c'est une représentation abstraite des connaissances et des activités qui régissent un domaine applicatif particulier facilitant la compréhension du système modélisé. Elle permet de décrire l'ensemble des concepts et technologies autour de la gestion des modèles. La phase de spécification est particulièrement importante dans une approche MDE et représente une partie conséquente du cycle de développement. Cela permet aux développeurs de se concentrer sur le comportement souhaité du système, sans se soucier de la manière de l'implémenter. La phase d'implémentation est alors démarrée en fin de cycle, une fois que la spécification est achevée et validée. La génération partielle du code, bas niveau, à partir de la spécification permet également de réduire le temps et donc les coûts de développement.

L'approche MDE vise à générer tout ou une partie de l'application à partir des modèles. Ce qui permet en soi d'augmenter la productivité tout en optimisant la compatibilité entre les systèmes grâce à la réutilisation à grande échelle des modèles normalisés. Ceci permet de simplifier le processus de conception, et de promouvoir la communication entre les individus et les équipes travaillant sur un système *via* une normalisation des terminologies et les meilleures pratiques utilisées dans le domaine du génie logiciel.

Dans ce qui suit, nous allons aborder les concepts et les principes de MDE tout en explicitant son extension appelée Architecture Orientée Modèles proposée par l'OMG (Object Management Group).

2.1 Les principes de l'ingénierie dirigée par les modèles (MDE)

Ces dernières années, de nombreuses organisations s'intéressent à l'ingénierie orientée modèles car celle-ci est une approche qui fournit les bases pour l'utilisation des modèles afin de mieux appréhender, concevoir, construire, déployer, maintenir et modifier un système. Il s'agit d'une évolution avantageuse pour plusieurs raisons. Premièrement, l'approche MDE encourage l'utilisation, d'une façon efficace, des modèles dans le processus de développement logiciel, puis

elle soutient la réutilisation des meilleures pratiques lors de la création d'un système. Ses principaux objectifs sont la portabilité, l'interopérabilité et la réutilisabilité à travers la séparation des aspects dépendants de la plateforme et des aspects plus abstraits indépendants de l'application [MM03].

Cette architecture a été introduite et définie par l'OMG et s'inscrit ainsi dans la continuité de l'approche orientée objet, en augmentant le niveau d'abstraction en un niveau où un autre jeu de concepts et de relations est utilisé, c.-à-d. modèle. Le modèle est une condition essentielle à la réalisation d'une architecture solide ainsi qu'une bonne compréhension du système à développer. C'est une spécification souvent partielle des fonctionnalités, de la structure et du comportement d'un système. Ce modèle peut être considéré comme une abstraction d'un système modélisé sous la forme d'un ensemble de faits, exprimé par la suite dans un langage de modélisation clairement défini pour formuler ce que l'on appelle un métamodèle.

Plusieurs approches MDE ont vu le jour. La plus populaire de ces approches est l'architecture dirigée par les modèles de l'OMG. Celle-ci a été étendue et ne se limite pas aux modèles (p. ex. UML) mais plutôt met au centre de la démarche en génie logiciel les modèles et non pas les programmes. Il s'agit de l'architecture orientée modèles (MDA).

2.2 Architecture orientée modèles (MDA)

L'architecture MDA est un moyen de structuration et de gestion de l'architecture logicielle d'une organisation où les modèles sont utilisés à grande échelle. En effet, elle permet de définir une structuration des lignes directives pour les spécifications exprimées en tant que modèles. Elle est supportée par des outils automatisés et des services pour, à la fois, définir les modèles et faciliter les transformations entre les différents types de modèles. En d'autres termes, MDA permet de séparer la spécification des fonctionnalités du système étudié de la spécification de son application sur sa plateforme d'implémentation. Elle offre la possibilité de concevoir des modèles indépendants de toutes plateformes ou environnements d'implémentation. L'approche MDA fournit un moyen au travers des modèles pour orienter la compréhension, la conception, le déploiement, l'exploitation et la maintenance du système étudié.

Dans le cycle de la mise en oeuvre de l'architecture MDA, trois groupes de modèles [JMJM06] ont été spécifiés comme illustré dans la figure I.4 :

- les modèles indépendants des traitements CIM (*Computation-Independent Model*) : le modèle le plus abstrait dans l'approche MDA. Il représente le contexte et le but du modèle sans aucune complexité de contrôle.
- les modèles indépendants de la plateforme PIM (*Platform-Independent Model*) : le modèle

décrit le comportement et la structure de l'application indépendamment de la plateforme mise en oeuvre.

– les modèles spécifiques à la plateforme PSM (*Platform-Specific Model*) : le modèle ne peut pas être exécuté mais il contient toutes les informations nécessaires, concernant une plateforme spécifique, que les développeurs peuvent utiliser pour l'implémentation.

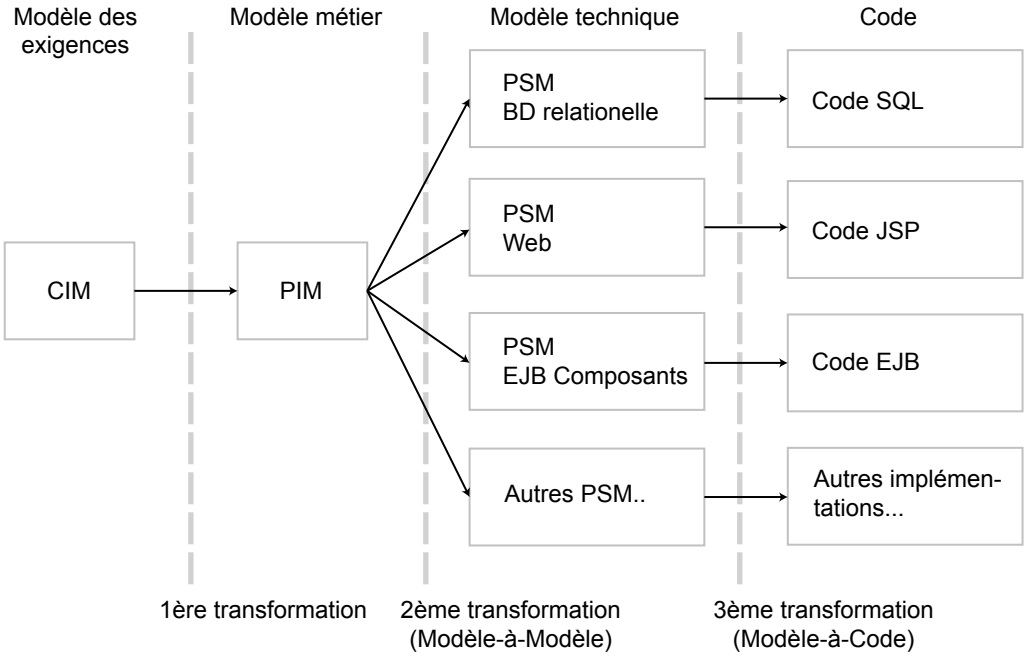


Figure I.4 – Cycle de développement élaboré par l'approche MDA

Afin d'implémenter de telle architecture, on peut utiliser un outil qui permet de modéliser l'application globale en incluant les logiques métiers et l'interface utilisateur dans un format générique. Nous pouvons sélectionner, après, une plateforme de déploiement dans laquelle cette application sera générée. Pour mieux appréhender la signification et le rôle de chacun de ces modèles, prenons l'exemple d'une entreprise qui désire mettre en place un processus d'appréciation de ses clients. Dans ce cas, le CIM symbolise le représentant qui contacte les clients par mail, par téléphone ou par courrier. Alors qu'au niveau du modèle PIM, le mécanisme sera réalisé par l'application logiciel mais une partie sera définie en termes d'information et de comportement du modèle (c.-à-d. le mail et le téléphone seront automatiques, mais le courrier devra être envoyé par une personne). Quant au PSM, on identifie pour implémenter le mail par exemple, que les développeurs utiliseront le package JMail et les diagrammes de séquences correspondants seront représentés. Il est important de noter que les étapes et les règles de transformation entre les modèles (c.-à-d. CIM au PIM et PIM au PSM) sont précisées à tra-

vers des transformations réversibles afin de garantir la traçabilité des décisions prises lors de la modélisation, et de réutiliser les modèles pour d'autres plateformes et d'autres systèmes.

Les modèles et l'implementation de MDA se basent principalement sur le standard UML [11] (*Unified Modeling Language*) [JMJM06]. Ce langage de modélisation est un standard également soutenu par l'OMG et permet la modélisation, la visualisation et la spécification des systèmes logiciels. Il est important de noter que les approches UML et MDA ont un grand avantage pour le développement des services Web composés. D'une part, les modèles de haut niveau permettent d'isoler le développeur des changements des technologies utilisées. En effet, dans le cas d'une évolution du langage de programmation, le nouveau code peut simplement être régénéré à partir des modèles de base, sans aucune autre intervention du développeur. D'autre part, les modèles de haut niveau tels que UML peuvent être convertis en toute implémentation de processus. Ils permettent également aux développeurs de générer le code dans différents langages, tels que BPMN ou Java. Enfin, les transformations peuvent être bidirectionnelles. Par conséquent, il est possible de générer des modèles de haut niveau à partir du code afin d'aider à la compréhension de la composition. Par exemple, dans le travail de [DGW08], un nouveau profil d'UML appelé UML-S, pour supporter la composition des services Web en se basant sur MDA a été défini. UML-S fournit un ensemble de stéréotypes et de contraintes ayant pour but d'étendre les métamodèles UML. UML-S vu comme un modèle PIM, est une vue du système indépendante de sa plateforme. Par conséquent, il est adapté pour une utilisation dans différentes plateformes du même type. En outre, il convient de noter qu'un service Web composé appelle simplement d'autres services et les rend interactifs. Cependant, il y a moins de programmation à effectuer en comparaison avec les services Web habituels. Ce qui implique que le code d'un service Web composite peut être entièrement généré à partir des modèles graphiques d'UML-S. En d'autres termes, les modèles UML-S (PIM) peuvent être directement traduis en code qui peut être déployé sans intermédiaire (PSM).

Toutefois, il existe d'autres modèles de conception graphique utilisés par MDA. Un de ces modèles est les Réseaux de Petri (RdPs). Cet outil graphique a été utilisé dans différents domaines d'application. Il est utilisé pour la conception des systèmes répartis et des systèmes temps réels [Oli05], pour la conception de la logique métier des applications et pour la conception des systèmes d'information basés sur les services Web [BHLJ04]. Les auteurs [RA05] utilisent une variante étendue des RdP à savoir les RdPs colorés qui est un outil adapté pour la modélisation des modèles CIM, PIM et PSM de complexité différente. L'utilisation de cette classe est justifiée, tout d'abord, parce qu'en général les RdPs permettent de modéliser des

11. UML, http ://www.uml.org/

propriétés telles que la concurrence et la synchronisation, et également parce qu'il est possible de définir à l'aide de cet outil les règles de transformation entre les différents modèles (de CIM au PIM et de PIM au PSM). [RA05] introduit une extension des RdPs, il présente également les aspects de structuration de plusieurs RdPs qui sont définis pour soutenir la traçabilité des changements et de la réutilisabilité, ainsi que la portabilité et l'interopérabilité des modèles.

Les principes de MDE sont tout à fait applicables au développement des services Web puisqu'il s'agit de composants logiciels. De plus, l'approche MDA apporte des atouts certains pour le développement des systèmes logiciels en général et des services composés en particulier. Tout d'abord le développement des modèles PIM épargne aux développeurs les changements et les évolutions liés aux technologies utilisées, puisque la composition des services est un domaine très actif où les technologies évoluent considérablement. Dans ce cas, l'approche MDA présente l'avantage de proclamer l'utilisation des modèles qui ne seront pas affectés par ces changements. Seules les règles de transformation de PIM au PSM et de PSM au code sont susceptibles de changer. De nouveaux codes peuvent être ainsi régénérés à partir des modèles originaux avec peu d'intervention de la part des développeurs. L'utilisation des modèles abstraits présente également l'avantage de pouvoir générer différents types de codes en passant éventuellement par différents PSM. Il est ainsi possible de générer le code BPEL, WSCI ou XLANG à partir du même PIM.

3 Composition et interaction de services

Actuellement, un nombre considérable d'entreprises utilise les services Web pour mettre à disposition leurs données et leurs informations *via* un réseau. Cependant, l'intégration de ces services dont l'objectif est de mettre en oeuvre la collaboration inter-entreprises, constitue une problématique conséquente dans la composition de services. Les organismes de recherche et les industriels ont manifesté un grand intérêt pour le développement d'une solution adéquate pour réaliser et faciliter cette tâche. L'architecture présentée dans la section 1.2 est une infrastructure basique largement suffisante pour implémenter une interaction simple entre un client et un service Web. Cependant, si l'implémentation de la logique métier d'un service fait appel à d'autres services, il est nécessaire, dans ce cas, de combiner les fonctionnalités de ces services. Le service résultant de cette combinaison ou de cette interaction est ce que l'on appelle un service composé. Le processus de développement de tels services est appelé la composition de services. Ce processus peut être réalisé en composant des services élémentaires ou des services composés [DS05]. Par conséquent, le service composé, à son tour, est récursivement définit par agrégation

des services élémentaires et/ou des services composés. Les règles de la composition décrivent comment les différents services peuvent être composés en un seul service final en cohérence avec la requête de l'utilisateur. En particulier, elles précisent l'ordre dans lequel les services sont utilisés et les conditions pour lesquelles ils seront invoqués. Néanmoins, la composition de services est une tâche très complexe, qui a reçu beaucoup d'intérêt pour soutenir les échanges B2B [BG07], et qui reste difficile à réaliser comme nous le verrons plus en détail dans cette section.

Dans ce qui suit, nous allons présenter une classification des différentes approches de composition de services en termes de workflow. Nous allons également adresser les différents types de composition de services. Nous allons citer ainsi certains langages utilisés par chaque approche. La figure I.5 présente une classification des approches de composition de services Web. Les éléments définis dans cette classification sont détaillés dans ce qui suit.

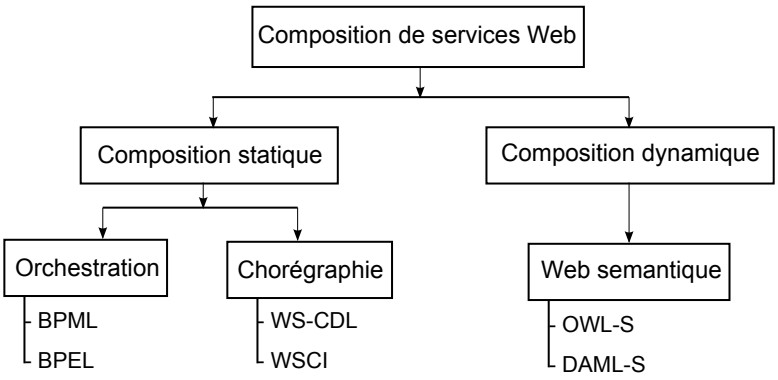


Figure I.5 – Les approches de la composition de services Web

3.1 Composition statique de services

La composition statique est réalisée durant la conception. Les composants à utiliser sont choisis et liés ensemble et sont finalement compilés et déployés. Cela devrait fonctionner tant que l'environnement du service (p. ex. disponibilité) n'a pas changé. Par contre, si de nouveaux services sont disponibles ou remplacés par d'autres, des incohérences seront relevées. Dans ce cas, il est inévitable de changer ou modifier l'architecture du système et de le relier à nouveau à d'autres services, ou, dans le pire des cas, de changer la définition du processus et concevoir à nouveau le système. Cependant, la composition statique peut être très restrictive et les composants doivent être adaptés automatiquement aux changements inéluctables.

Au sein de l'architecture SOA, le mécanisme de la composition de services décrit comment les différents services peuvent être composés en un service final pour satisfaire la demande de

l'utilisateur. Comme évoqué en détail dans [Hu03], les approches basées sur le workflow sont classées en deux catégories : les méthodes d'orchestration et les méthodes de chorégraphie. Ces deux approches ont comme objectif commun la collaboration entre les processus métiers impliquant plusieurs services autonomes, où chaque service assume un rôle bien défini et forme une relation spécifique avec les autres services.

Il est important de noter que : WSDL, UDDI et SOAP permettent la description et la découverte des services. Néanmoins, ils ne prennent pas en considération l'intégration et la composition automatique et dynamique des services. En effet, ils spécifient seulement les services et les opérations qui devront être effectuées, sans respecter l'ordre des messages de spécifications échangés entre les services. Différents langages de spécification de flux ont été proposés pour décrire l'ordre des messages échangés entre les services, par exemple BPEL4WS, XLANG, WSFL et DAML-S.

3.1.1 Méthodes d'orchestration

Cette approche combine les services disponibles en ajoutant un coordinateur central (comme un chef d'orchestre) qui a la responsabilité de gérer et invoquer les services Web. Comme indiqué dans l'exemple de la figure I.6, le chef d'orchestre ou le médiateur est une agence de voyage. Quand un client souhaite réserver un voyage, il contacte directement l'agence de voyage qui s'occupe de toutes les interactions avec les autres services en arrière plan, à savoir, le service *compagnie aérienne*, le service *hôtels* et le service *location de voiture*. Une fois que le client recevra toutes les informations concernant son voyage, il procède au paiement en invoquant, *via* le médiateur, le service *paiement*.

L'orchestration peut donc être considérée comme une construction d'un processus automatisé en différents services qui définissent les étapes des échanges dans ce processus. L'orchestration décrit donc les interactions entre plusieurs services afin de fournir une nouvelle fonctionnalité en se basant sur le nouveau service composé. Parmi les langages utilisés pour l'orchestration dans la composition de service on cite :

- WSFL (*Web Service Flow Language*) proposé par IBM et qui permet de spécifier comment mettre en oeuvre un modèle de processus métier en utilisant l'architecture des services Web. En effet, WSFL s'intègre avec UDDI et WSDL pour la sélection des services Web. Les fournisseurs de service doivent bien décrire leurs services afin qu'ils soient classifiés pour traiter une activité spécifique dans le processus métier. Ce dernier est décrit en utilisant WSDL et représenté par un modèle de flux XML, qui définit les activités implémentées sous le format des services Web, et définit également les séquences de flux de données entre les activités.

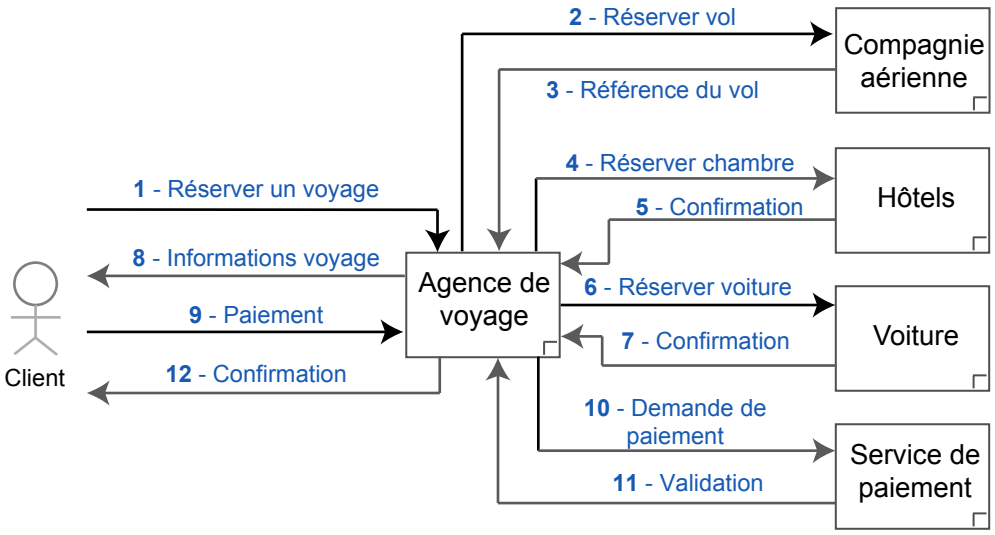


Figure I.6 – Exemple d'orchestration : Réservation d'un voyage.

- XLANG est un autre langage de composition de flux développé initialement par Microsoft pour la création des processus métiers et l'interaction entre les fournisseurs de service. La spécification fournit un support pour le séquencement, le parallélisme et le contrôle conditionnel de flux. Ce langage utilise WSDL pour décrire l'interface du service du processus. Il étend le langage WSDL en ajoutant des capacités de comportement, où un comportement définit la liste des actions qui appartiennent au service et l'ordre dans lequel elles doivent être exécutées. Cet ordre est définit à travers les processus de contrôle (p. ex. séquence, while loop).

- BPEL4WS, qui est une dérivée à la fois de WSFL et XLANG, est un langage permettant de décrire les interactions des services Web comme étant un processus métier. La définition de ce dernier implique la spécification précise du comportement visible des messages échangés de chaque partie impliquée dans ce processus, sans révéler leur mise en oeuvre interne. De cette façon, BPEL4WS définit comment les interactions multiples des services sont coordonnées afin d'atteindre les objectifs métiers, ainsi que l'état et la logique nécessaire pour cette coordination. Il suppose que les services sont décrits par un langage basé sur XML comme WSDL.

3.1.2 Méthodes de chorégraphie

La chorégraphie consiste à concevoir une coordination décentralisée des applications dans laquelle il n'y a pas de privilège (c.-à-d. serveur) accordé à un des acteurs de cette approche mais plutôt, un ensemble de participants qui échangent des messages et effectuent un traite-

ment. Suivant cette approche, l'ensemble des activités est achevé comme la composition des interactions *peer-to-peer* (P2P, qui est un modèle de réseau informatique proche du modèle client-serveur mais où chaque client est aussi un serveur) entre les services participants. La figure I.7 présente un exemple d'une chorégraphie. Comme on peut le remarquer, les interactions ne sont pas centralisées par l'agence de voyage mais plutôt, chacun des services exécute son rôle indépendamment des autres services. Plus précisément, une fois que le client effectue le paiement en interrogeant le service *paiement*, ce dernier va automatiquement notifier les services *agence de voyage* et *location de voiture* sans qu'aucun de ces deux services ne demande de confirmation.

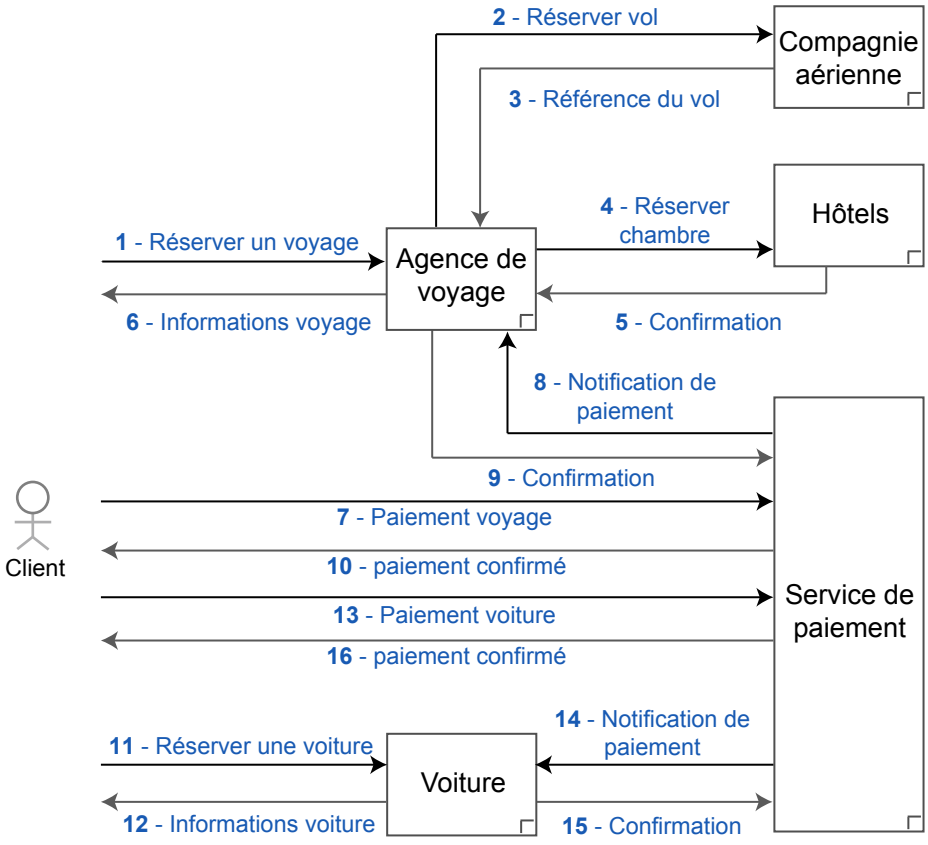


Figure I.7 – Exemple de chorégraphie : Réservation d'un voyage.

La chorégraphie, contrairement à l'orchestration, ne se base pas sur un coordinateur central, mais plutôt définit la conversation qui devrait être entreprise par chaque service Web participant (p. ex. le chorégraphe pour la danse). Plus précisément, la chorégraphie définit les règles et les interactions de collaboration entre deux ou plusieurs services. De ce fait, la chorégraphie est

plus collaborative que l'orchestration et permet à chaque service appelé de décrire son rôle dans la composition ou l'interaction. Les langages définis pour cette approche sont décrits dans ce qui suit.

- WS-CDL (*Web Service Choreography Description Language*) [KBR+05] est un langage utilisé pour la chorégraphie et permet la spécification des protocoles peer-to-peer où chaque partie est autonome, et n'a pas d'hiérarchie vis-à-vis des autres parties. Plus précisément, WS-CDL est une collection d'activités qui peut être effectuée par un ou plusieurs participants. Il existe trois types d'activités dans WS-CDL, l'activité de contrôle de flux (p. ex. séquence, parallèle et choix), les activités de l'unité de travail et les activités de bases. Les activités de contrôle de flux sont similaires aux constructions basiques du contrôle de flux dans les langages de programmation typiques et impératives. Ces activités correspondent au séquence, flux ; alors que les activités *Switch* et *Pick* correspondent au BPEL4WS. L'unité de travail, quant à elle, décrit une exécution répétée possible ou conditionnelle d'une activité. Le dernier type d'activité de WS-CDL décrit les points dans la chorégraphie quand un rôle exécute une action qui n'affecte pas le reste de la chorégraphie.

- WSCI (*Web Service Choreography Interface*) est un langage de description basé sur XML qui permet de chorégraphier le flux des messages entre les services Web. Ce langage décrit le flux des messages échangés par un service Web dans un processus particulier. Il permet notamment de décrire les messages collectifs échangés parmi les services Web qui interagissent, en fournissant une vue globale du processus impliquant les multiples services Web. WSCI est un langage basé sur une structure qui, permet de traiter l'observable externe plutôt que la définition interne du comportement du service, qui est exprimée en termes des dépendances temporaires et logiques parmi les messages échangés. Cependant, il n'expose aucune forme de la sémantique et par conséquence ne facilite pas le processus de la composition dynamique.

L'environnement des services Web est de nature flexible et dynamique. Le nombre de fournisseurs de services augmentent constamment et de nouveaux services deviennent disponibles. Idéalement, les processus d'un service devraient être capables de s'adapter d'une manière transparente aux changements de l'environnement et de s'accommoder aux exigences des utilisateurs avec un minimum d'intervention du concepteur. La tâche la plus difficile reste à composer les services dynamiquement. En particulier, lorsque la fonctionnalité qui ne peut pas être réalisée par les services existants est nécessaire, et l'invocation d'un autre service s'avère indispensable pour effectuer cette fonctionnalité. Les deux compositions, statique et dynamique, dépendent du paramètre temporel notamment lors de la composition. Elles sont équivalentes respectivement à la composition au moment de la conception où les services composants sont connus au préalable (pour la composition statique) et à la composition au moment de l'exécution où les

services sont découverts à cet instant (pour la composition dynamique).

3.2 Composition dynamique de services

La composition dynamique de services Web est la technologie clé de l'implémentation de l'architecture SOA. Cependant la sélection de services représente un réel problème dans ce type de composition. La complexité de la sélection des services inclue trois facteurs principaux [QXQY09]. Le premier concerne le grand nombre de changement dynamique des instances des services ayant des fonctionnalités similaires et qui pourraient être disponibles pour composer un service. Le deuxième facteur concerne les différentes possibilités d'intégration, des composants d'une instance de service, au processus d'un service complexe. Tandis que le troisième facteur correspond aux diverses exigences de performance d'un service complexe (c.-à-d. introduction de critères de la qualité de service tels que le délai, le prix et la fiabilité).

Le nombre de services et de scénarios de collaboration étant important, l'analyse manuelle de ces services, dont l'objectif est d'achever les requêtes de l'utilisateur, reste difficile et au-delà de la capacité humaine. Il est donc important de définir un mécanisme de composition de services automatique. Pour cela, le Web sémantique a été introduit comme étant au dessus du Web actuel afin d'associer à chaque donnée un sens bien défini, pouvant être interprété par un ordinateur.

Dans le reste de cette section, nous allons présenter brièvement le Web sémantique et le principe de l'ontologie en citant quelques langages utilisés pour la définition des ontologies. Nous allons par la suite découvrir la problématique de la concurrence des services dans la composition où la sélection de services s'impose.

3.2.1 Web sémantique

Dans la composition dynamique de services Web, un agent logiciel devrait chercher, au moment de l'exécution d'un processus, tous les services susceptibles de répondre à la requête de l'utilisateur. Cette recherche devrait accéder à des ressources Web par le contenu plutôt que par des mots-clés. Par conséquent, ce contenu devrait être compréhensible et interprétable par cet agent logiciel. C'est la fonction du Web sémantique. En effet, ce dernier désigne un ensemble de technologies visant à rendre le contenu des ressources du Web accessible et utilisable par les programmes et agents logiciels [BLHL01]. Les efforts engagés pour la création du Web sémantique prennent de l'ampleur afin d'accéder à des ressources Web par le contenu plutôt que par des mots-clés. On constate l'intérêt du Web sémantique lorsque deux fournisseurs de services sont identiques, mais que chacun définit ses propres fonctions d'une façon différente alors que

l'on parle de la même fonction. Dans ce cas, même si chaque fournisseur code son information d'une manière différente, c'est la couche sémantique qui rend possible les interactions et les échanges de données entre, d'une part, ces fournisseurs et d'autre part les utilisateurs du service. De plus les données échangées peuvent être comprises par l'ordinateur.

En informatique, les ontologies ont été adoptées dans le domaine de l'intelligence artificielle pour faciliter le partage et la réutilisation de connaissances [Fen03]. Ce sont des modèles conceptuels capturant et rendant explicite le vocabulaire utilisé dans des applications sémantiques. En d'autres termes, une ontologie est une collection de services Web qui partagent le même domaine d'intérêt ou plus précisément la même sémantique. Les ontologies permettent la description sémantique des services Web. L'utilisation des ontologies pour modéliser le comportement et le processus des services dynamiques a suscité beaucoup d'intérêt de la part des chercheurs et des industriels [DKL+05, DS05, KC10, FMS+05]. Par exemple, dans le travail [KC10], une approche a été proposée pour automatiser la tâche de la composition. Plusieurs langages, tels que DAML, DAML + OIL, et OWL, ont été développés pour spécifier les services composés. Nous rappelons les définitions de quelques uns de ces langages.

- DAML-S (DARPA Agent Markup Language for Web Services) [DAM11] est un langage basé sur XML et permet de faciliter l'automatisation des tâches des services Web. Il met à disposition aux fournisseurs de services un ensemble de moyens et de constructions basés sur le langage de balisage pour la description des propriétés et des capacités des services Web d'une façon non ambiguë et interprétable par l'ordinateur.

- OWL-S (*Ontology Web Language*) [fDSC11] est basé sur l'ontologie pour fournir la description des services Web. Il s'agit d'une révision du DAML+OIL, langage d'ontologie intégrant les apprentissages tirés de la conception et l'application du DAML + OIL. OWL-S, qui est une extension de DAML-S, vise à mettre en place un cadre sémantique pour les services Web. Il dépend des ontologies pour réaliser la découverte, l'invocation, la composition et l'interopérabilité automatique des services.

3.2.2 Sélection de service

Avec la prolifération des services Web en tant que solution fondamentale d'intégration d'application d'entreprise, la QoS (Quality of Service) est de plus en plus importante pour les fournisseurs de services. Toutefois, en raison des caractéristiques dynamiques et imprévisibles des services Web, il n'est pas toujours évident de fournir la QoS désirée pour les utilisateurs de service. Par ailleurs, différentes applications avec des exigences variées en termes de critères de QoS seront en compétition pour les ressources du réseau et du système, tels que la bande

passante et le temps de traitement. Néanmoins, une QoS améliorée apportera un avantage concurrentiel pour les fournisseurs de services. Pour cela, il faudrait prendre en considération les critères de la QoS qui ne se limitent pas seulement au *prix* et au *coût* mais incluent d'autres critères [LJL+11] comme la *disponibilité*, la *fiabilité*, etc.

Lors de la composition dynamique des services, l'introduction des critères de QoS dans le processus de sélection de services a fait l'objet de plusieurs travaux de recherche [ZBD+03, QXQY09]. En effet, la sélection de service est une étape inévitable dans ce type de composition. Notamment, lorsqu'un processus métier ou une fonctionnalité, que l'on aimerait exécuter, est composé de plusieurs tâches pour lesquelles différents services Web en concurrence susceptibles de répondre à la requête se présentent. Par conséquent, le processus métier global pourrait être présenté sous forme d'une arborescence où plusieurs plans d'exécution peuvent être appliqués. Dans ce cas, une approche par optimisation devient nécessaire.

Différentes techniques et méthodes d'optimisation ont été déployées en fonction des systèmes étudiés pour l'optimisation de la composition de services Web [PFC08, YM08, LNZ04]. Parmi ces approches, celles basées sur l'optimisation multi-objective, mono-objective, globale ou locale. Nous citons ici quelques travaux qui ont traité cette partie de l'optimisation dans la composition de services.

Dans [QXQY09], les auteurs proposent une approche d'optimisation multi-objective en définissant un algorithme d'optimisation de la sélection de service Web. Cet algorithme est basé sur la colonie de fourmis pour la composition dynamique des services Web. Ce problème est transformé dans un premier temps en un problème d'optimisation multi-objective en prenant en considération les contraintes des utilisateurs. De plus, puisqu'il est basé sur MOACO (Multi-Objective Ant Colony Optimization), l'algorithme permet d'optimiser simultanément les différents paramètres de la QoS des workflows. Dans ce sens un ensemble de solutions optimales (ensemble Pareto) est calculé.

Dans [PFC08], le problème de composition basée sur la QoS a été formulé comme un problème d'optimisation NP-difficile. Les auteurs appliquent des techniques d'optimisation métaheuristiques à ce problème, plus précisément la méthode d'optimisation *recherche Tabu* ainsi qu'un algorithme génétique hybride. Ils ont également comparé ces techniques avec d'autres propositions dont l'algorithme génétique de base.

Une autre technique est utilisée pour l'optimisation de la sélection dans la composition de services dans les travaux de [YM08]. Cette approche est basée sur une combinaison de la programmation linéaire, le raisonnement par cas et les algorithmes génétiques. Le schéma proposé a permis la réduction des coûts grâce à la réutilisation des compositions existantes. Ce schéma a montré une efficacité importante en termes de réduction du temps par rapport à

d'autres travaux basés sur la programmation linéaire.

Nos travaux de recherche prennent en compte lors de la sélection de services des critères de QoS et de l'optimisation pour sélectionner le meilleur service pour exécuter une tâche donnée. En effet, nous avons considéré certains critères de la QoS dans la modélisation et l'évaluation de performances de notre système. Le chapitre 3 et le chapitre 4 détaillent cette partie de la sélection et l'optimisation de la composition de services.

4 Conclusion

Dans ce chapitre, nous avons présenté les technologies de base encadrant notre domaine de recherche. Il s'agit des principes et des technologies de la composition de services, particulièrement l'architecture orientée service, les LBS, et les langages de description de service. Ensuite, nous avons brièvement présenté les deux approches orientées ingénierie et orientées modèle, en citant quelques travaux réalisés dans la littérature. Enfin, nous avons présenté les deux types de composition de services Web à savoir la composition statique et la composition dynamique, ainsi que les langages et approches liés à chacun des types de la composition.

Dans ce qui suit, nous allons nous intéresser aux approches orientées modèles pour la vérification formelle et l'évaluation de performances de la composition de services. Nous proposerons dans ce sens une analyse sous forme d'une classification des approches et des modèles existants dans la littérature. Nous allons également positionner notre travail par rapport à l'existant.

Chapitre II

Approches orientées modèles pour la composition de services

Ce chapitre présente une étude de l'état de l'art des approches orientées modèles pour la composition de services avec comme critère de comparaison leurs fonctions et leur mode de fonctionnement. La section 1 est consacrée à notre proposition d'une classification de ces approches en s'appuyant sur les fonctions réalisées par chaque modèle (p. ex. la modélisation, l'évaluation de performances, etc.). Dans la section 2, nous allons présenter les approches orientées modélisation et génération de code. Les approches pour la vérification formelle sont adressées dans la section 3. La section 4 est consacrée aux approches pour l'évaluation de performances de la composition de services. Enfin, dans la section 5, nous terminons par l'analyse de l'existant et le positionnement de notre travail par rapport aux recherches développées dans ce domaine.

1 Classification des approches

La composition de services permet de définir et faciliter le processus d'intégration et de collaboration entre plusieurs services intra-entreprise ou inter-entreprise. Cette technologie émergente a eu beaucoup d'intérêt autant dans le domaine de la recherche que dans le domaine de l'industrie. Comme cela a été présenté précédemment, de nombreux langages ont vu le jour au cours de ces dernières années tels que XLANG, WSFL, BPEL ou encore WSCI. Ces approches reposent ainsi sur des concepts de programmation et négligent de ce fait l'étape de spécification qui doit prendre place au début du cycle de développement. De plus, les langages proposés manquent de formalisme permettant la vérification et la validation de la composition de services. Il n'est donc pas possible d'appliquer directement les méthodes mathématiques sur ces langages, rendant ainsi la vérification difficile. En effet, les processus de validation et de vérification visent à assurer la conformité des systèmes aux exigences requises et le respect des caractéristiques

de conception attendues. Ces processus apportent tout leur intérêt pour conclure sur la bonne description du modèle, et pour s'assurer du bon fonctionnement du système étudié avant sa réalisation et son implémentation. Par conséquent, il est indispensable de prendre en compte cette étape en se basant sur d'autres approches et modèles pour la spécification formelle de la composition de services.

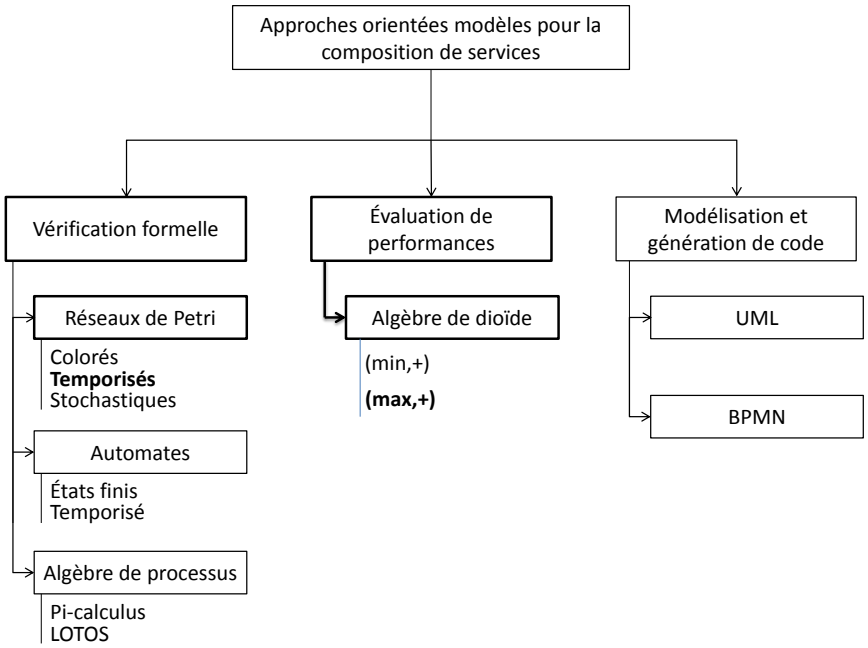


Figure II.1 – Une classification des approches orientées modèles pour la composition de service

Les approches orientées modèles pour la composition de services peuvent être classées en trois types selon leurs fonctions. La Figure II.1 présente la classification de ces approches qui peuvent être classés en trois grandes familles, les approches orientées vérification formelle, les approches orientées modélisation et génération de code, et les approches orientées évaluation de performances.

L'approche proposée dans ce travail appartient à deux familles, les approches orientées évaluation de performances et les approches orientées vérification formelle illustrée en gras dans la figure II.1. Nous présentons quelques travaux de recherche menés par rapport à chacune des familles proposées.

2 Approches orientées modélisation et génération de code

Les approches appartenant à cette famille proposent de modéliser la composition de services sous forme d'un processus métier (*Business Process*). Les standards BPMN (*Business Process Management Notation*) et UML ont été utilisés pour modéliser la composition de services. Ils présentent l'avantage d'être basés sur l'utilisation des notations graphiques permettant une utilisation et une compréhension simples et exhaustives. Ces notations sont suffisamment expressives pour permettre la génération de code à partir des modèles graphiques.

2.1 Langage de modélisation unifié (UML)

UML est un langage standardisé par l'OMG permettant de modéliser un système selon différents points de vue, statique et dynamique. La structure statique permet de modéliser un système en utilisant des objets, attributs, opérations et relations. La structure dynamique permet de modéliser le comportement dynamique d'un système en montrant les interactions entre les objets ou les changements d'états au sein d'un objet. Les diagrammes d'états et d'activités font partie de cette catégorie.

UML, en particulier le diagramme d'activités, a été adapté pour la modélisation de la composition de services. Par exemple, dans [AGGI04], un profil UML a été introduit pour la modélisation de processus métier à l'aide des diagrammes d'activités. Un nouveau langage basé sur le diagramme d'activité UML 2.0 pour la modélisation de la composition de services a été présenté dans [BGB05]. En utilisant ce langage, le code BPEL peut être généré à partir des modèles de la composition. Dans [DGW08], UML-S (UML pour les Services) a été proposé pour modéliser la composition de services. Les auteurs ont également introduit des règles de transformation des diagrammes UML-S en BPEL.

2.2 Notation de gestion des processus métiers (BPMN)

BPMN [1] (*Business Process Management Notation*) a été développé par BPMI (*Business Process Management Initiative*) et maintenu à présent par le groupe OMG/BPMI [Gro08]. C'est un standard pour la modélisation des processus métiers ainsi que des processus des services Web. Son but principal est de fournir un cadre permettant de décrire un processus d'une manière commune à tous les utilisateurs et ce indépendamment de l'outil utilisé. Il permet de définir principalement trois objets de base, tâche, événement, et branchement. Une tâche est un élément granulaire caractérisé par un début et une fin. Un événement identifie un état particulier dans

1. BPMN, http ://www.bpmn.org

le processus (c.-à-d. début, intermédiaire, fin). Un branchement sert à représenter les routages entre les flux d'entrées et les flux de sorties (p. ex. branchement exclusif, parallèle, conditionnel, etc.).

BPMN se compose d'un diagramme - appelé le diagramme de processus métiers (BPD). Ce dernier a été conçu pour être facile à utiliser et à comprendre, mais offre également la possibilité de modéliser des processus métiers complexes. Dans [Whi04], les auteurs ont réalisé une comparaison entre les diagrammes de processus métiers et les diagrammes d'activités UML pour la modélisation de vingt et un workflows patterns. En effet, il y a beaucoup de similitude entre les deux diagrammes en termes de notations et de représentations, mais BPMN fournit un certain nombre d'avantages par ses fondements mathématiques qui sont conçu pour se transformer facilement en langages métiers. De même, BPMN peut être traduit en UML et fournit ainsi un moyen de modélisation solide face aux systèmes de conception UML.

BPMN peut être utilisé pour la composition de services grâce à sa possibilité d'offrir des notations permettant de modéliser les interactions entre les services sous forme d'un processus métier afin générer le code du service composé exprimé en langage de composition tels que BPEL. Plusieurs outils ont été développés pour convertir un modèle du BPMN en BPEL en utilisant les règles de transformation présentées dans [A.04, ODtHvdA07]. Dans [SZS10], une approche pour intégrer un ensemble d'exigences de qualité dans les modèles BPMN a été proposée. L'objectif est de considérer les exigences de la qualité au niveau des processus métiers, afin de choisir les services appropriés. Pour maintenir le même niveau d'abstraction que pour les modèles de processus, les auteurs ont met l'accent sur les exigences de la qualité de service étroitement liées aux préférences des clients.

3 Approches orientées vérification formelle

Des modèles avec une sémantique formelle ont été proposés afin de vérifier que le processus de composition de services fonctionne correctement. Ces approches sont généralement utilisées pour la spécification et la vérification formelle des systèmes complexes. En effet, ces méthodes formelles permettent de simuler et vérifier le comportement de la composition de services au moment de la conception. Cependant, la vérification permet de détecter et de corriger les éventuelles erreurs le plutôt possible et avant toute implémentation. Il existe plusieurs méthodes et modèles pour transformer les descriptions du processus de composition de services (OWL-S, DAML-S et BPEL4WS) en descriptions formelles en utilisant des méthodes telles que les RdPs ou l'algèbre de processus. Nous rappelons, dans ce qui suit, quelques travaux développés dans

la littérature au sujet de l'utilisation de ces méthodes formelles.

3.1 Réseaux de Petri

Les réseaux de Petri ont été introduits par C.A. Petri [Pet62] pour la modélisation des systèmes concurrents. Un des avantages clé de cet outil est la façon naturelle dans laquelle de nombreux aspects fondamentaux, à savoir l'aspect mathématique et l'aspect conceptuel, des systèmes concurrents sont identifiés. Ceci a contribué à l'élaboration d'une théorie riche des systèmes concurrents basés sur les RdP [JMW+07]. Leur facilité de modélisation conceptuelle, due à une notation graphique facile à comprendre, a par ailleurs fait des RdP l'outil de choix dans de nombreuses applications. Les RdPs sont très populaires dans le domaine de gestion des processus métiers (BPM) en raison de la variété des processus de flux de contrôle qu'ils peuvent modéliser [WvdAP+03, vdA98].

Plusieurs études de recherche ont été développées pour modéliser la composition de services en utilisant les RdP. Par exemple, Hamadi et al. dans [HB03] ont introduit une algèbre basée sur les RdP pour composer les services web. Cette composition est basée sur les flux de contrôle. Dans [OVvdA+07], la transformation des instructions BPEL en RdP est réalisée. Les auteurs ont utilisé la classe des RdPs étiquetés. Les modèles obtenus ont été utilisés pour vérifier les processus BPEL par le biais des outils BPEL4PNML et WofBPEL.

Dans [YK04a] un framework pour la conception et la vérification de la composition de services basée sur les RdPs est proposé. Cet outil est utilisé pour visualiser, créer et vérifier les processus BPEL (deux interfaces permettant de présenter d'une part le modèle en RdP et d'autre part le modèle équivalent en BPEL). Dans [HSS05] une description complète et officielle de la sémantique des RdPs pour BPEL est présentée. Par ailleurs les auteurs présentent leur analyseur BPEL2PN qui peut transformer automatiquement les processus BPEL en un modèle RdP. Par conséquence, les auteurs peuvent utiliser une variété d'outils de vérification des RdP, à titre d'exemple *VisualObjectNet++* [2], afin d'analyser et de vérifier automatiquement les processus BPEL présentés sous forme de modèles RdP.

Les auteurs de [NM02a] ont défini la sémantique d'un ensemble d'instructions DAML-S en RdP. En se basant sur cette sémantique, ils ont décrit la décomposition de services à l'aide du formalisme RdP, et ont mis en oeuvre un outil pour décrire et vérifier automatiquement la composition de services. Dans [NM02b], deux approches sont prises en compte lors de la composition de services. La première se base sur le passage des descriptions ou modèles OWL-S aux modèles RdP afin de les analyser davantage. Pour cette approche, la description OWL-S

2. VisualObjectNet++, http ://www.r-drath.de/Home/Visual_Object_Net++.html

est automatiquement transformée, le développeur utilise alors le résultat pour automatiser les tâches de vérification de la composition. La deuxième approche se base sur les transformations au Prolog.

Les RdPs constituent un framework pour la modélisation de la composition de services. La facilité de la compréhension, l'utilisation de ses notations graphiques, ainsi que sa puissance de vérification et d'analyse formelles à travers un ensemble de bases et techniques mathématiques, justifient le choix de l'utilisation de cet outil pour des problématiques de modélisation et de vérification de la composition de services. De plus, différents logiciels (CPN Tool [3], TINA [4], etc.) pour la vérification du comportement des modèles RdP sont disponibles, permettant ainsi de détecter les erreurs liées à la modélisation avant l'implémentation du système.

3.2 Automates

Un automate est un modèle très connu dans le domaine de la spécification formelle des systèmes complexes et notamment des systèmes à événements discrets. Il est composé d'un ensemble de noeuds représentant les états, un ensemble d'actions et un ensemble d'arcs étiquetés décrivant les transitions entre les états. Son comportement est dirigé par un mot fourni en entrée, l'automate passe alors d'un état à un autre pour exécuter ce mot. On retrouve les automates dans la modélisation de processus, dans les protocoles de communications, dans l'étude des langages formels et dans la spécification des processus de composition de services. Différentes classes d'automate ont été proposées pour modéliser le comportement d'un système, tels que les automates entrée/sortie (E/S) et leurs nombreuses variantes [LT89], les automates temporisés [AD94], les automates à états finis ou les automates (max,+) [KLB09].

Les automates temporisés ont été introduits pour modéliser formellement le comportement des systèmes temps-réels. Il s'agit d'une extension des automates à états finis où des contraintes temporelles sont prises en compte. Ces automates utilisent un nombre fini d'horloges à valeurs réelles qui peuvent être réinitialisées et dont les valeurs augmentent uniformément dans le temps. Chaque horloge mesure le temps écoulé depuis sa dernière initialisation. Ces horloges peuvent être utilisées pour garder le passage d'un état à l'autre. C'est-à-dire, une transition n'est validée que si la contrainte temporelle associée à l'état satisfait la valeur actuelle de l'horloge. Plusieurs outils ont été développés pour la vérification des modèles (automates) en prenant en compte ces contraintes temporelles, par exemple UPPAAL [LPY97] et KRONOS [Yov97].

Les automates à E/S ont été introduits pour modéliser les systèmes distribués [LT87]. Fon-

3. CPN Tool, http ://cpntools.org/
4. TINA, http ://projects.laas.fr/tina

damentalement, un automate à E/S est un automate dont l'ensemble des actions sont partitionnées entre les actions internes et externes (c'est à dire les actions d'entrée et de sortie) utilisées pour communiquer avec un environnement composé d'autres automates à E/S.

Dans le contexte de notre travail, les automates ont été utilisés pour décrire, spécifier et vérifier la composition de services. Par exemple, dans [DPC+05], les auteurs proposent une traduction des descriptions des services Web, définis en BPEL-WSCDL, en automates temporisés. Ces transformations, réalisées d'une manière automatique, sont par la suite vérifiées par UPPAAL. Dans [PZWQ06], des règles ont été proposées pour traduire un sous-ensemble de BPEL en automates temporisés et ainsi vérifier le processus BPEL à l'aide du UPPAAL. Dans [MGI06], une approche permettant la description des processus OWL-S en automates à états finis a été proposée dont l'objectif est de sélectionner les meilleurs services composés en fonction des requêtes des utilisateurs exprimées aussi par des processus OWL-S .

3.3 Algèbre des processus

Les algèbres de processus sont des familles de langages formels permettant de modéliser les systèmes logiciels, en particulier les systèmes distribués et concurrentiels. Elles constituent un outil pour une description du plus haut niveau d'interaction, de communication et de synchronisation entre les processus. Elles définissent des règles algébriques permettant d'une part, l'analyse et la manipulation des descriptions des processus, et d'autre part, le raisonnement formel des équivalences entre les processus. Un tel raisonnement est très utile pour déterminer par exemple si un service peut remplacer un autre dans la composition, ou de vérifier simplement la redondance des services. Le fondement sémantique des algèbres des processus est basé sur des systèmes de transition étiquetés, à savoir les automates. De nombreuses variantes de ces langages formels ont été proposées dans la littérature. Nous citons par exemple, CSP (Calculus of Sequential Processes) [Hoa83] proposé par Hoare, CCS (Calculus of Communicating Systems) [Mil89] introduit par Milner et utilisé pour évaluer l'exactitude qualitative des propriétés d'un système comme le blockage, et LOTOS (Language Of Temporal Ordered Systems) [BB87] qui a été standardisé par ISO. Ces algèbres sont des formalismes bien étudiés permettant la vérification automatique de certaines propriétés des comportements des systèmes.

Les algèbres de processus ont été utilisés pour modéliser et vérifier la composition de services [BBG07, Dum10]. Par exemple, dans [SBS04], les auteurs préconisent l'utilisation des algèbres de processus pour décrire, composer et vérifier les services Web, avec un accent particulier sur leurs interactions. Une étude de cas a été présentée dont l'objectif est de montrer comment CCS peut être utilisé pour spécifier et composer les services Web en tant que processus, afin de

valider les propriétés liées à une composition correcte de services Web. D'autres algèbres plus avancées que CCS sont préconisées afin de considérer également les échanges de données au cours des interactions et compositions dynamiques de services Web. Par exemple, dans [Fer04], une transformation bidirectionnelle a été définie pour passer du BPEL au LOTOS et vice versa. Un avantage de cette traduction est qu'elle inclut la gestion des exceptions et permet la vérification des propriétés temporelles.

Dans [DGW08] les auteurs ont proposé la modélisation de la composition de services en utilisant LOTOS. Le modèle développé et validé par l'outil CADP [5] (Construction and Analysis of Distributed Processes) est utilisé pour la vérification de la composition. Plus précisément, des règles ont été présentées pour permettre la transformation entre les principales structures de contrôle des workflows d'un service composé et LOTOS avant de procéder à la vérification formelle de la composition à l'aide de CADP.

4 Approches orientées évaluation de performances

Dans les deux sections précédentes, nous avons présenté les deux grandes familles des approches orientées modèles pour la composition de services, les approches orientées vérification formelle et les approches orientées génération de code. Les approches appartenant à la première famille utilisent des langages formels pour spécifier et vérifier la composition de services. Les approches appartenant à la deuxième famille utilisent des modèles abstraits sans considérer la manière dont ils sont implémentés. L'utilisation combinée des outils de ces deux familles permet de réduire les coûts de développement et faciliter la réutilisation de ces modèles. Malgré les avantages que présentent ces approches, en particulier leur simplicité et leur haut degré d'expressivité, l'évaluation de performances de la composition de services reste une tâche difficile.

Cependant peu de travaux existent dans la littérature pour l'évaluation de performances de la composition de services. Par exemple, dans [Dua11], une approche basée sur l'utilisation du Network Calculus [LBT01] a été proposée pour l'évaluation de performances des différents scénarios de la composition de services. L'objectif est de sélectionner le service composant qui satisfait une certaine QoS (p. ex. délai, coût).

Dans [cbsmW10], une approche basée sur l'utilisation des RdP et l'algèbre (max,+) pour la modélisation de la composition de services. Les auteurs s'intéressent plus précisément à la modélisation des conflits par ces deux outils formels. Ils ont définit une politique de routage permettant la résolution et l'arbitrage des conflits structurels.

5. http ://cadp.inria.fr/

L'algèbre de dioïde est un autre formalisme dédiée à la modélisation et à l'étude de performances des systèmes dynamiques à événements discrets (SDED). Elle permet d'obtenir un modèle théorique linéaire dans une structure algébrique non usuelle contre un modèle non forcément linéaire dans l'algèbre usuelle. Cette caractéristique fait de cette algèbre un outil pertinent et bien adéquat pour la modélisation et l'analyse des phénomènes complexes tels que la synchronisation, la concurrence et le parallélisme. Il est important de noter que cette algèbre peut être combinée avec d'autre outils formels, tels que les graphes à événements discrets permettant ainsi d'offrir plus d'avantages et plus de puissance pour une modélisation et une analyse complètes en allant jusqu'à la commande et l'optimisation. Comme mentionné précédemment, parmi les classes de l'algèbre de dioïde, on cite les modèles (max,+) et (min,+) qui, permettent d'étudier certains systèmes dont les comportements évoluent dans un espace d'état discret. Il a été justifié à plusieurs reprises que la théorie des systèmes linéaires dans l'algèbre des dioïdes est une théorie riche en propriétés pour l'étude et l'analyse des SDED ainsi que l'évaluation de leurs performances [CM89, FGGJJ92, NSMACBW09, HOvdW05, HBL06].

5 Discussion

Notre travail dans cette thèse s'inscrit dans le cadre des approches orientées modèles pour la vérification formelle et l'évaluation de performances de la composition de services. Cette approche est basée sur l'utilisation des deux outils formels, les RdP pour modéliser et vérifier la composition de services et l'algèbre (max,+) pour l'évaluation de performances. Il est important de noter que la fonction principale des RdP est la vérification formelle, d'ailleurs dans la littérature cet outil est utilisé principalement pour vérifier formellement les processus dans la composition de services. Toutefois, il est possible d'évaluer certaines performances de ces processus en utilisant les RdP grâce à un ensemble de techniques mathématiques associées à cet outil, mais dans certains cas, en particulier dans le notre, cet outil présente une limite ou une complexité dans cette représentation mathématique (la non linéarité des équations mathématiques). Pour faire face à cela, nous associons les RdPs à l'algèbre (max,+). Ceci présente un avantage crucial pour une étude complète des processus dans la composition de services. En effet, cette algèbre est dédiée à la détermination et à l'analyse des propriétés de certains systèmes dont le comportement peut être représenté sous forme de modèles mathématiques (max,+) linéaires. Notre objectif est d'utiliser cette théorie ainsi que ses techniques de calcul et d'analyse pour représenter et étudier le comportement de la composition de services par une représentation d'état dans l'algèbre des dioïdes.

Compte tenu, d'une part, de la puissance de l'algèbre (max,+) dans la modélisation et l'évaluation de performances des SDED, et d'autre part, du manque des travaux de recherche au sujet de l'utilisation de cette algèbre pour l'évaluation des processus de la composition de services, notre objectif est d'apporter notre contribution dans ce domaine. A travers ce travail, nous décrivons les comportements des interactions et des échanges entre les services par des équations mathématiques, et aussi d'évaluer certaines propriétés quantitatives de ces comportements. Ces derniers font apparaître des situations de conflits, aussi bien dans le processus que dans le modèle RdP associé, compte tenu de plusieurs facteurs dont la nature fluctuante des interactions entre les services, le choix de services et d'autres facteurs d'instabilité. Face à cette situation, notre objectif est d'étendre l'algèbre (max,+) pour la modélisation des SDED dont les comportements sont gérés par des conflits et des choix. Nous montrons, dans la suite de ce mémoire, comment ces comportements sont intégrés dans les équations (max,+) et quelles sont les solutions proposées pour gérer et arbitrer les différentes situations complexes rencontrées. Nous montrons également que notre approche, contrairement aux approches proposées dans la littérature, permet d'évaluer les performances de la composition de services afin de valider les processus de composition de service.

L'étude d'un système (notamment l'évaluation de ses performances) peut se faire en plusieurs phases qui se distinguent selon l'outil de modélisation utilisé et la nature du modèle obtenu. Dans notre cas, deux phases d'évaluation de performances sont envisageables. La première s'applique dans le cas de la représentation du système sous forme d'un modèle graphique obtenu à partir de l'outil RdP. Elle permet d'une part, d'étudier le comportement structurel du système, et d'autre part de vérifier les propriétés qualitatives du système. La deuxième phase se consacre à l'évaluation analytique qui se fait à partir des modèles mathématiques décrivant le système. Cette méthode permet d'effectuer une analyse algébrique et une validation des modèles. L'analyse et l'évaluation de performances du système correspondent à l'analyse et à l'évaluation de ses caractéristiques telles que, les composants du système, le coût, le temps de traitement, etc. Plus précisément, nous nous intéressons à la modélisation des processus de composition de services vus comme des *workflow patterns*. Ces derniers représentent des modèles automatisés des processus métiers, en particulier les services Web. Ils sont utilisés pour représenter les tâches récurrentes dans la modélisation des processus métiers. Les différents workflows patterns sont représentés par des modèles RdP ensuite par des modèles (max,+) afin de vérifier et d'évaluer les performances de la composition de services.

6 Conclusion

Dans ce chapitre, nous avons présenté les approches orientées modèles pour la composition de services existants dans la littérature. Une classification de ces approches ainsi qu'une brève description de notre approche est réalisée. Nous avons justifié le choix des outils de modélisation, de vérification et d'évaluation adoptés pour notre étude.

Le chapitre suivant sera consacré à un ensemble de définitions relatives à la théorie des dioïdes, aux RdPs et workflow patterns. Nous mettons aussi l'accent sur la description des workflow patterns par des modèles RdP et par des équations (max,+). En d'autres termes, nous développons et analysons des modèles graphiques et mathématiques pour la vérification formelle et l'évaluation de performances des processus de composition de services.

Chapitre III

Spécification et modélisation formelles de la composition de services

Dans ce chapitre, nous allons décrire les workflows patterns comme outil de base pour la description et la représentation des séquencements de tâches dans la composition de services. Par la suite, nous allons utiliser conjointement les deux modèles formels à savoir les RdP et l'algèbre (max,+) afin d'analyser et d'étudier les performances des scénarios de composition de services. Ce chapitre est donc scindé en trois sections. Dans la section 1, nous rappelons les définitions des patterns et des workflows, nous citons également quelques travaux de recherche développés à ce sujet. Ensuite, nous présentons dans la section 2 un rappel des RdP et de l'algèbre (max,+). Dans la section 3, nous décrivons le comportement graphique de chaque pattern par un modèle RdP. Ensuite, nous présentons le comportement analytique de chaque modèle RdP (autrement dit de chaque pattern) en équations mathématiques dans l'algèbre (max, +). Nous finissons la section par un exemple d'illustration de la composition de services.

1 Introduction

1.1 Patterns et Workflows

La notion de *pattern* a été introduite par Christopher Alexander [AIS77] comme étant le concept d'un langage permettant de décrire les relations entre des modèles spécifiques. A l'origine, l'utilisation des patterns est concentrée sur l'architecture informatique, mais le concept a une porté générale et a été largement répandu dans d'autres domaines. Cependant, il a eu plus d'impact dans le domaine des technologies de l'information où les patterns ont été utilisés pour le classement des principaux concepts, notamment dans la conception des systèmes, dans l'analyse métier, dans la conception des processus métiers, dans l'architecture des systèmes et dans l'intégration des applications d'entreprises.

Un *workflow* est définit comme étant la modélisation, la gestion informatique et automatique de l'ensemble des tâches à accomplir ainsi que des différents acteurs impliqués dans la réalisation d'un processus métier. En d'autres termes, un workflow décrit le circuit de validation, les tâches à accomplir entre les différents acteurs d'un processus, les délais, les modes de validation, et fournit à chacun des acteurs les informations nécessaires pour la réalisation de sa tâche. Les modèles de workflow se concentrent sur la façon dont le travail est effectué pour atteindre les objectifs définis par l'organisation. Cette technique permet de définir comment la tâche, les informations et les documents sont transmis d'un utilisateur à un autre au sein d'une organisation.

L'utilisation des workflows est devenue un atout majeur fournissant un avantage concurrentiel, comme étant un moyen pour réduire les coûts, automatiser le processus, et réduire le temps de réponse. Correctement implémentées, les applications basées sur les workflows permettent aux entreprises de restructurer et de rationaliser les processus métiers, ce qui justifie la croissance et l'ampleur de l'intérêt des workflows ces dernières années.

Cela fait plus de dix ans que l'initiative des workflows patterns se focalise sur l'identification du noyau architectural des constructions dans la technologie du workflow [vdAHKB00]. L'objectif initial était de délimiter les exigences fondamentales, qui se posent lors de la modélisation des processus métiers de façon récurrente, et de les décrire d'une façon impérative. Une approche basée sur les patterns a été considérée pour décrire ces exigences, du fait qu'elle offre un moyen indépendant à la fois du langage et de la technologie [YGN09]. Cette approche permet d'exprimer les caractéristiques essentielles des exigences sous une forme suffisamment générique pour permettre son application à une grande variété de produits.

Les workflow patterns ont fait l'objet de plusieurs études au cours de ces dernières années, notamment en termes d'exhaustivité et de précision. En vue d'assurer leur pertinence, une évaluation complète a été menée et dont les résultats ont été présentés dans [RAvdAM06]. En plus de cette évaluation, l'un des principaux objectifs de cette étude était de réviser la description de chacun des patterns et de les décrire d'une façon plus formelle afin de lever toute ambiguïté potentielle qui pourrait exister. Ceci a été réalisé en utilisant les RdP colorés [Jen03] qui ont permis la description des opérations de chaque pattern d'une façon formelle.

L'étude élaborée dans [vdAHKB00] a confirmé l'application des vingt patterns originaux qui continuent à être largement utilisés dans la littérature. Elle a permis également d'identifier vingt-trois nouveaux patterns, certains d'entre eux sont basés sur le raffinement des modèles d'origine, et d'autres sont le résultat d'une évaluation critique des facteurs supplémentaires qui sont pertinents aux perspectives des flux de contrôle. Un examen complet des patterns, dans [RAvdAM06], a été réalisé pour quatorze produits différents, notamment pour les systèmes de

workflow (Staffware, COSA, WebSphere MQ, CSR, iPlanet, SAP Workflow et FileNet), pour les systèmes de gestion de documents (FLOWEr), pour les langages de modélisation des processus métiers (BPMN, UML 2.0 Diagramme d'activité et EPCs), et pour les langages d'exécution des processus métiers (BPEL4WS, WebSphere BPEL, Oracle BPEL et XPDL).

1.2 Composition de services basée sur les patterns

Dans le domaine de l'informatique orientée services, ou SOC - *Service-Oriented Computing*, l'une des plus importantes fonctions est de créer des services à valeur ajoutée, c.-à-d. un service composé, qui est le résultat de la composition de plusieurs services Web existants. Un service composé peut être modélisé, tout comme un processus métier, avec la logique interne entre ses composants en utilisant un langage de modélisation des processus métiers adapté pour les services Web, par exemple BPEL, WSCI, etc. Le service composé est obtenue à travers l'invocation et la coordination des services composants [JRGM04, HYJY08].

Bien que l'accent a été mis sur les systèmes workflow, il est clair que les patterns étaient applicables dans un sens beaucoup plus large et ont été utilisés pour examiner les capacités des langages de modélisation des processus métiers tels que BPMN et le diagramme d'activité UML, les langages de composition de services Web tels que WSCI et les langages d'exécution des processus métiers tels que BPML, XPDL et BPEL. La modélisation des processus métiers a fait l'objet de plusieurs recherches, et n'a pas été consacrée ou limitée à l'automatisation de ces processus en utilisant les systèmes d'information. Les modèles workflow comprennent différentes notations, langages et outils logiciels. Notamment les RdPs [vdA96], les diagrammes d'états-transitions, les notations BPMN et les diagrammes d'activités UML ou UML-S [DGW08] comme une extension des patterns supportés par UML 2.0.

Aalst et al. ont proposé plus de quarante patterns dans [RAvdAM06]. Parmi ces patterns, une variété peut être utilisé pour modéliser la communication et l'interaction entre les services Web en termes de contrôle de flux. Par exemple, les auteurs de [JRGM04] définissent quatorze patterns parmi les vingt originaux et les utilisent pour la composition de services Web. Ils ont également définit de nouveaux patterns en composant ceux existants. Ils se sont basés sur la QoS comme critère de sélection dans la composition de services, et ont fourni les agrégations des services résultants en termes de deux critères de la qualité de service à savoir le coût et le prix. Dans [HYJY08], les auteurs ont défini une approche basée sur les workflow patterns, plus précisément sur neuf patterns, pour adapter la composition de services lors de l'exécution du processus métiers. En effet, les processus métiers sont composés de services Web qui sont exécutés dans un environnement volatile où les paramètres de la qualité de services participants

peuvent être changés durant l'exécution du processus, notamment, quand il s'agit de remplacer un service qui échoue lors de l'exécution d'une tâche donnée. Ce changement a un coût non négligeable dont on doit tenir compte. Par conséquent, l'approche proposée dans [HYJY08] prend en compte la valeur de l'information changée lors de l'exécution d'un processus et donc de la composition de services. Dans [BPG06], les auteurs ont définit une extension des workflows patterns en patterns transactionnels pour la composition de services Web. Différents patterns ont été utilisés pour spécifier et orchestrer une composition de services dite fiable et flexible, tout en définissant des règles appropriées afin d'éviter les incohérences qui peuvent être engendrées dans la composition de patterns.

Un système est dit concurrentiel si plusieurs de ses traitements sont réalisés en parallèle, et qu'une interaction entre les processus actifs est susceptible de se produire en cours d'exécution. En termes de composition de services, un service composé représente l'agrégation de plusieurs services. Ces services sont susceptibles de s'exécuter au même instant, où plusieurs services sont candidats pour l'exécution d'une tâche donnée, où encore, quand un service doit être choisi (le plus pertinent possible) pour l'exécution d'une tâche parmi un ensemble de tâches. De ce fait, la composition de services peut être considérée comme un système concurrentiel. Cependant, l'analyse et le traitement de ces systèmes où des contraintes de choix, de concurrence et de conflits s'imposent, nous incitent à introduire d'autres facteurs, d'autres politiques et d'autres arguments, pour faire face à ces situations et répondre au mieux aux exigences des utilisateurs finaux. Dans cette étude, nous introduisons la qualité de service (QoS) comme facteur adéquat permettant de prendre une décision face à des situations de concurrence citées auparavant. Nous revenons sur ce facteur de la QoS en détails dans la section suivante. En effet, les services composés sont souvent basés sur des services distribués. Ce qui introduit des facteurs liés aux communications réseaux, d'où l'utilisation de la notion de qualité de service.

1.3 Critères de la QoS dans la composition de services

Dans la composition de services Web, les services peuvent être sélectionnés et intégrés afin de générer un workflow pour satisfaire les requêtes des utilisateurs. Cette composition dépend de différents critères, et plusieurs services proposent les mêmes fonctionnalités pour satisfaire la requête d'un client, un problème d'optimisation se pose dans le but de définir les services à sélectionner. Autrement dit, sélectionner les services qui répondent au mieux aux exigences en termes de QoS. Afin de satisfaire cet objectif, plusieurs approches ont été proposées dans la littérature [QXQY09, PFC08]. Cependant la plupart de ces approches se basent sur l'optimisation locale de la QoS ou l'optimisation mono-objective, par conséquent les contraintes définies

par les utilisateurs, et qui sont globales de manière générale, sont difficiles à respecter dans la plus part des cas. De ce fait, la sélection de service qui s'appuie sur l'optimisation globale de la QoS est un problème NP-difficile [PFC08]. Toutefois, il est crucial de prendre en compte, à ce stade, les critères de la QoS lors de la composition de services. En effet, cela nous permet, non seulement de définir les services les plus adéquats pour réaliser un scénario donné mais, en plus d'étudier et de représenter le système en se rapprochant de son exécution dans un environnement réel. Nous allons donc définir, dans ce qui suit, les différents critères de la QoS que nous avons pris en compte dans la sélection de services abordée dans ce travail.

1.3.1 Définition des critères de la QoS

Nous allons définir dans cette section les différents critères de la QoS qui peuvent être déterminés ou évalués au moment de l'exécution d'un service Web, et qui impactent d'une façon optimale la sélection et la composition de service. Nous distinguons deux types de critères :

1. Le critère prédéterminé est connu avant l'exécution du service,
2. Le critère déterminé à la phase de l'exécution reflétant des informations dynamiques qui dépendent des dernières conditions de l'invocation d'un service donné.

Ces critères sont pris en considération dans la modélisation par les RdPs. Plus précisément, lors de la modélisation d'un processus, ou d'un scénario, si les critères de la QoS en question sont déterminés avant l'exécution du service, le modèle RdP décrivant le comportement de ce scénario est un modèle sans conflits et sans contraintes de choix.. Dans ce cas le scénario est représenté par un graphe d'événements temporisés (GET) qui est une classe des RdP. Dans le deuxième cas, si les critères de la QoS ne sont pas connus par avance, des contraintes de choix et de conflits s'imposent, ce qui conduit à considérer, lors de la modélisation, des classes plus compliquées mais adéquates des RdP. L'introduction de ces classes dédiées fera l'objet de la section 2.1.

Dans ce qui suit, nous définissons les quatres critères que nous considérons dans notre étude de sélection de services. Bien entendu, des critères supplémentaires peuvent être intégrés et considérés en fonction du système étudié.

- Durée

La durée d'exécution d'un service élémentaire s noté par $q_t(s)$. Elle est mesurée comme le délai entre le moment où une requête est envoyée et le moment où le résultat est reçu par le demandeur. Elle est calculée par l'équation (III.1) :

$$q_t(s) = t_{process} + t_{trans} \tag{III.1}$$

Où :

- $t_{process}$ est le temps nécessaire au service pour exécuter sa tâche. Ce paramètre est fourni par la description du service.
- t_{trans} est le temps de transmission (en général ce temps dépend de l'état du réseau). Il est estimé à partir des exécutions précédentes du même service.

- Coût

Un coût est affecté à chaque service élémentaire s. Ce coût est à la charge du demandeur de service. Il est noté par $q_p(s)$.

- Disponibilité

La disponibilité d'un service élémentaire s, représente la probabilité pour qu'un service soit accessible. Elle est notée par $q_d(s)$ et est exprimée comme suit :

$$q_d(s) = \frac{T_a}{\theta} \tag{III.2}$$

Où T_a $(0 \leq T_a \leq \theta)$ représente le temps global pour lequel un service était disponible pendant un intervalle de temps $[0, \theta]$ (θ est une constante définie par l'administrateur du service en fonction du type du service).

- Fiabilité

La fiabilité d'un service s est la probabilité pour que la réponse à une requête soit correctement donnée dans les délais prévus (ces délais sont fournis dans la description du service Web). La valeur de la fiabilité d'un service est calculée à partir des données historiques des invocations précédentes du même service. Elle est exprimée comme suit :

$$q_f(s) = \frac{N_c}{K} \tag{III.3}$$

Où N_c est le nombre de fois où le service a été exécuté avec succès dans le délai prévu et K est le nombre total d'invocations.

Considérant les critères de QoS définis ci-dessus, le vecteur de la QoS d'un service s_i est défini comme suit :

$$q(s_i) = (q_p(s_i), q_t(s_i), q_d(s_i), q_f(s_i)) \tag{III.4}$$

Ces critères peuvent être appliqués pour évaluer les services composés. Pour cela, la QoS de chaque service composé (c.-à-d. représenté par un pattern) sera calculée en fonction des agrégations de chaque critère défini avant. Ces agrégations seront exprimées dans ce qui suit.

1.3.2 Définition du score de services composants

Comme évoqué précédemment, nous associons à chaque service qui est candidat dans un processus de composition de services, le vecteur QoS (voir l'équation (III.4)). Pour calculer le score de chaque service, nous prenons en considération les préférences de l'utilisateur lors de l'établissement de sa requête. Pour ce faire, ces préférences sont exprimées sous forme de poids associé à chaque critère. Le score est donc exprimé comme suit :

Score$(s_i) = \sum w_j q_{ij}$, où $w_j \in [0, 1]$ est le poids assigné au $j^{ème}$ critère de la QoS du service s_i, sachant que $\sum w_j = 1$ et q_{ij} est la valeur du critère j.

Comme dit précédemment, nous nous intéressons à l'exécution de tout le plan (c.-à-d. le schéma de l'ensemble des services qui peuvent exécuter les différentes tâches). En effet, au lieu d'exécuter une unique tâche, l'exécution se fait plutôt au niveau du plan global de la composition. Le calcul du score se fait donc pour le plan global d'exécution et non pas pour chaque tâche. Cependant, nous allons nous intéresser uniquement à la sélection globale des services. Pour ce faire, nous introduisons la matrice de qualité, notée **Q** par la suite, pour tous les services concurrents, susceptibles d'exécuter toutes les tâches d'un plan. Cette matrice est définie comme suit :

$$\mathbf{Q} = \begin{pmatrix} Q_{11} & Q_{12} & Q_{13} & Q_{14} \\ Q_{21} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ Q_{n1} & Q_{n2} & Q_{n3} & Q_{n4} \end{pmatrix} \tag{III.5}$$

Q_{ij} est la valeur du $j^{ème}$ critère de la qualité de service du $i^{ème}$ plan. Nous signalons que Q_i (où i est la $i^{ème}$ ligne de **Q**) représente le vecteur de la QoS d'un plan d'exécution et non pas celui du service élémentaire. En effet, Q_i est obtenu à partir des agrégations des critères de tous les services candidats dans le plan et qui sont présentés dans un ordre bien défini.

Afin de définir le score de chaque service, nous devons prendre en considération les pondérations associées à chaque critère de service. Ces pondérations expriment les préférences des utilisateurs. Pour cela, nous appliquons la méthode de pondération additive simple [CCS08], qui se déroule en deux phases à savoir la phase de mise à l'échelle et la phase de pondération.

- Phase de mise à l'échelle

Un critère est dit négatif, quand la valeur de la qualité de service baisse si sa valeur augmente. Cela peut concerner par exemple le coût et le temps d'exécution. En effet, plus le coût d'un service est haut plus sa qualité est faible. Cela s'applique également pour le critère temps d'exécution ; plus le temps est grand plus la qualité du service est faible. Un critère est dit

positif si lorsque la valeur de la qualité augmente, sa valeur augmente (p. ex. fiabilité). Nous avons donc défini deux équations en fonction du type du critère. Pour les critères négatifs nous appliquons l'équation (III.6) et pour les critères positifs nous appliquons l'équation (III.7).

$$V_{i,j} = \begin{cases} \frac{Q_j^{max} - Q_{i,j}}{Q_j^{max} - Q_j^{min}} & \text{si } Q_j^{max} - Q_j^{min} \neq 0 \ j = 1,2 \\ 1 & \text{si } Q_j^{max} - Q_j^{min} = 0 \end{cases} \tag{III.6}$$

$$V_{i,j} = \begin{cases} \frac{Q_{i,j} - Q_j^{min}}{Q_j^{max} - Q_j^{min}} & \text{si } Q_j^{max} - Q_j^{min} \neq 0 \ j = 3,4 \\ 1 & \text{si } Q_j^{max} - Q_j^{min} = 0 \end{cases} \tag{III.7}$$

Dans les équations (III.6) et (III.7), Q_j^{max} (resp. Q_j^{min}) correspond à la valeur maximale (resp. minimale) du critère dans la matrice de la qualité (III.5). Après cette phase, nous obtenons la nouvelle matrice suivante :

$$\mathbf{Q'} = \begin{pmatrix} V_{1,1} & V_{1,2} & V_{1,3} & V_{1,4} \\ V_{2,1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ V_{n,1} & V_{n,2} & V_{n,3} & V_{n,4} \end{pmatrix} \tag{III.8}$$

- Phase de pondération

Après la phase de mise à l'échelle, nous calculons le score du plan d'exécution *via* la phase de pondération. La formule de calcul est la suivante :

$$Score(p_i) = \sum_{j=1}^{4} (V_{i,j} \times w_j) \tag{III.9}$$

Avec $w_j \in [0,1]$ le poids assigné au critère de la QoS, et $\sum w_j = 1$.

Il est important de noter que la sélection de services se base principalement sur ces deux phases. Une fois les scores calculés, nous sélectionnons le plan ayant le score maximal. Dans le cas où, plusieurs plans ont le même score maximal, nous choisissons au hasard un parmi tous ces plans.

La sélection de services est effectuée en amont de l'étape d'exécution des modèles (c.-à-d. pattern). En effet, chaque requête est décomposée en un ensemble de tâches exécutées dans un ordre bien défini. A partir de cet ordre, nous générons un pattern puis nous sélectionnons les services qui permettent de répondre à chaque tâche du pattern en se basant sur leurs scores calculés comme annoncé précédemment. Ensuite nous optimisons les plans d'exécution possibles en choisissant le plan optimal.

Pour ce faire, une technique d'optimisation globale est appliquée et qui sera détaillée dans le chapitre 4. Après la génération du plan d'exécution et la sélection des services associés, nous définissons la durée nécessaire pour l'exécution de chaque tâche comme étant une temporisation associée à la place du RdP modélisant cette tâche. Signalons que les différentes temporisations des modèles RdP que nous proposons par la suite sont obtenues après la phase de sélection de service et de définition des scores. Pour chaque service sélectionné (c.-à-d. dont le score est maximal), nous retenons son temps d'exécution moyen ainsi que ses temps d'exécution dans le pire et le meilleur des cas.

2 Modélisation par les RdPs et l'algèbre (Max,+)

L'initiative d'introduire les RdPs dans la composition de services a été proposée dans [OVB+05]. En effet, les auteurs ont proposé des règles de traduction complètes et précises des processus BPEL en modèles RdP. Ces règles de traduction ont été d'ailleurs largement exploitées dans des outils tels que WofBPEL qui est un outil de vérification automatique des processus BPEL. Le travail développé dans [vdA96] se focalise et s'appuie sur les qualités des RdP comme outil adéquat pour une modélisation claire et précise des workflows. En effet, la spécification des workflows à l'aide des RdP permet d'aboutir à un résultat de modélisation rigoureux et fidèle aux problèmes modélisés dans ses moindres détails. Cela est principalement possible grâce aux différentes extensions et méthodes de modélisation graphique des RdPs. Un ensemble riche de techniques et d'outils mathématiques est associé aux RdPs, ce qui permet de proposer une étude complète de la modélisation jusqu'à la validation en passant par la vérification et l'évaluation de certaines propriétés qualitatives, telles que la faisabilité, l'invariance, la sûreté de fonctionnement, le non-blocage. Les RdPs constituent un formalisme indépendant de tout langage ou plateforme spécifique, leur association avec l'algèbre des dioïdes permet d'étendre l'étude de modélisation et d'analyse des systèmes en proposant d'autres techniques mathématiques complémentaires, qui s'ajoutent à celles des RdPs. L'utilisation combinée de ces deux outils permet d'avoir une analyse complète des systèmes modélisés en allant jusqu'à leur commande à des fins d'optimisation et d'amélioration. De plus, l'association de l'algèbre des dioïdes aux RdPs constitue un atout complémentaire permettant d'étudier et d'évaluer certaines propriétés quantitatives telles que la date d'occurrence d'un événement, ou le nombre d'occurrence d'un événement à un moment donné.

2.1 Réseaux de Petri

Les RdPs ont été utilisés pour modéliser et analyser toute sorte de processus avec des applications allant des protocoles, du matériel et des systèmes embarqués aux systèmes de production [SV90], d'interaction avec l'utilisateur, et des processus métiers [vdA96]. Ils ont montré beaucoup d'intérêts dans différents domaines notamment dans la composition de services Web [YK04b]. Les RdPs représentent un outil dédié à l'étude des SEDs [CM89] dont la dynamique est régie par divers phénomènes dont la synchronisation, la concurrence et le parallélisme.

La modélisation d'un système par les RdPs est une transition de la réalité vers un objet graphique et par la suite vers un modèle mathématique. Cette transition va permettre une étude plus large du système, avec l'objectif d'étudier son comportement, d'évaluer ses performances et enfin de vérifier et de valider ses propriétés. En utilisant les RdPs, la vérification formelle d'un système se fait à partir de sa représentation graphique, elle permet d'étudier son comportement structurel et d'en tirer des propriétés qualitatives (p. ex. vivacité, non-blocage, etc.). A partir des techniques mathématiques fournies par les RdPs, une évaluation analytique peut être faite, ce qui permet d'effectuer une analyse du système et de le valider à partir de ses modèles. Notons qu'il est possible de procéder à la vérification d'un modèle RdP par simulation tout en simulant l'évolution de son comportement par un outil de simulation numérique tel que, VisualObjectNet [1] ou TINA [2]. Cette simulation permet de vérifier si le modèle obtenu est cohérent par rapport au fonctionnement attendu du système modélisé.

Selon le système étudié, ainsi que sa complexité, et dans l'objectif d'étudier son comportement d'une façon concrète, d'évaluer et d'analyser ses performances, plusieurs classes des RdPs ont été proposées dans la littérature. Dans ce qui suit, nous présentons les différentes classes que nous utiliserons dans notre travail de recherche.

Un RdP est un graphe biparti composé de deux types de sommets : places (modélisant les conditions, les états, ou les événements) et transitions (modélisant la transition d'un état à un autre état ou d'un événement à un autre événement). Des arcs orientés pondérés reliant des places à des transitions, ou des transitions à des places. Un arc ne relie jamais deux sommets de même nature. Les places sont représentées par des cercles et les transitions par des rectangles ou des barres. Chaque place peut contenir un ou plusieurs jetons (modélisant, par exemple, la capacité d'un stock, la disponibilité d'une ressource) représentés par des points. L'évolution et la dynamique d'un système modélisé par un RdP sont représentées par les franchissements de ses transitions et le déplacement des jetons des places en amont vers les places en aval de chaque

1. VisualObjectNet++ : $http://www.r-drath.de/Home/Visual_Object_Net++.html$
2. TINA : http ://homepages.laas.fr/bernard/tina/description.php

transition franchie. Comme évoqué précédemment, un RdP est un modèle à la fois graphique et mathématique, dédié à la description des systèmes de type à événements discrets.

Définition 2.1.1 *Un réseau de Petri est un 5-uplet $\langle P, T, A, W, M_0 \rangle$, où :*

- *P est un ensemble fini de places*
- *T est un ensemble fini de transitions*
- *$A \subseteq (P \times T) \bigcup (T \times P)$ est un ensemble fini d'arcs*
- *$W : A \rightarrow \mathbb{N}$ est la fonction poids associée aux arcs où on assigne à chacun un entier positif qui indique le nombre de jetons nécessaires pour franchir une transition ou le nombre de jetons produits après le franchissement d'une transition.*
- *$M_0 : P \rightarrow \mathbb{N}$ est le marquage initial du modèle RdP qui correspond au nombre de jetons dans les places à l'état initial.*

Remarque 2.1.1 *Le marquage d'un réseau de Petri se définit par l'application $M : P \rightarrow \mathbb{N}$, telle que $\forall P_i \in P$, M_{P_i} ou $M(P_i)$ définit le marquage de la place P_i, c-à-d le nombre de jetons contenus dans la place P_i à un moment donné.*

Remarque 2.1.2 *Dans ce chapitre, nous ne nous intéressons pas à l'évolution dynamique des RdPs modélisant les comportements des patterns. Le marquage des places est donc nul.*

Pour la suite nous adoptons les notations suivantes :

- L'ensemble T_i° (resp. $^\circ T_i$) désigne l'ensemble des places en aval (resp. en amont) de la transition T_i.
- L'ensemble P_i° (resp. $^\circ P_i$) désigne l'ensemble des transitions en aval (resp. en amont) de la place P_i.

Définition 2.1.2 *Soit $R = \langle P, T, A, W, M_0 \rangle$ un réseau de Petri, une transition $T_i \in T$ est dite tirable ou franchissable si et seulement si :*

$$\textit{Pour tout } p_j \in {}^\circ T_i \textit{ ; On a } (p_j, T_i) \in A \textit{ et } M(p_j) \geq W((p_j, T_i))$$

Avec $1 \leq i \leq |T|$ et $1 \leq j \leq |{}^\circ T_i|$.

Où $|T|$ est le cardinal de l'ensemble des transitions T.

Dans un RdP toute transition franchissable, notée par la suite T_f, peut être tirée et son tir conduit à un nouveau marquage de ses places en amont et en aval.

Définition 2.1.3 *Soit $R = \langle P, T, A, W, M_0 \rangle$ un réseau de Petri. R n'est pas borné si :*

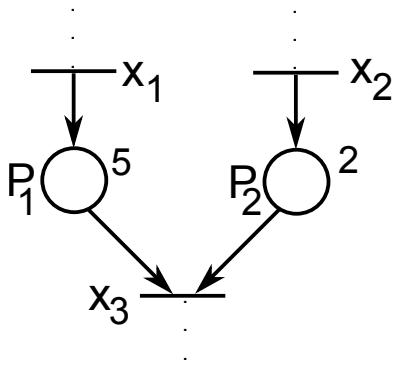


Figure III.1 – Graphe d'événements temporisés

$$\forall\, n \in \mathbb{N},\ \exists\, p \in P \text{ tel que } M(p) > n$$

Différentes classes des réseaux de Petri ont vu le jour pour répondre au mieux à la modélisation de divers systèmes. Chacune peut être utilisée pour modéliser un système spécifique, le concepteur choisit la classe la plus appropriée pour modéliser son système.

Une multitude de classes et de variétés de systèmes mettant en jeu des phénomènes de différentes natures proviennent de la littérature. Parmi ces phénomènes, nous citons ceux qui nous intéresse le plus dans notre étude de composition de services à savoir : le parallélisme, la synchronisation, l'exclusion mutuelle, le choix, le conflit, le séquencement, etc. Cette multitude de phénomènes a conduit au développement de différentes classes des RdPs pour répondre au mieux aux modélisations spécifiques de systèmes. Nous citons à titre d'exemple : les graphes d'évènements temporisés, les réseaux de Petri P-temporisés et les réseaux de Petri avec conflits. Dans ce qui suit, nous donnons les définitions de certaines de ces classes.

Définition 2.1.4 ***(Graphe d'Évènement)****. Un graphe d'événements est un réseau de Petri tel que toute place $p \in P$ a exactement une transition en amont et une transition en aval. Un graphe d'événements est dit P-temporisé, si à chaque place $p \in P$ on associe un temps $\Theta(p) = \tau_p$, où $\Theta : P \to \mathbb{N}$ est l'application qui associe à toute place p une temporisation τ_p. Cette dernière correspond à la durée du séjour d'un jeton dans la place p.*

Notons que les graphes d'événements ne permettent pas de modéliser certains phénomènes dont la concurrence, le choix, le conflit. Néanmoins, cette classe des RdPs est intéressante pour de nombreuses applications où il apparaît des phénomènes de synchronisation ou de parallélisme entre plusieurs processus [NSMACBW09, smMMM03].

Le réseau de Petri est un formalisme bien adapté à la modélisation des SED, plus particulièrement le GET (Graphe d'Évènements Temporisé). Comme nous le constatons à travers cet

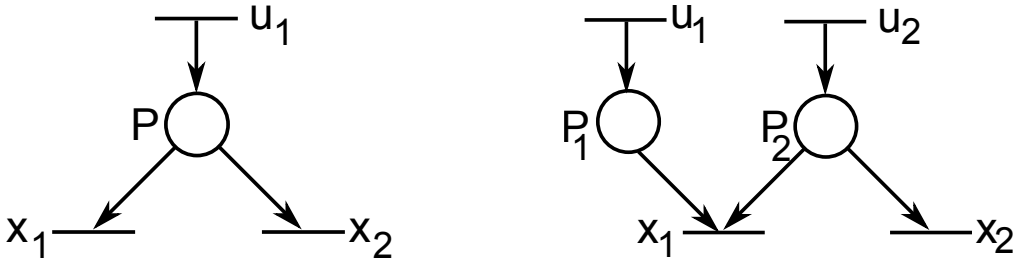
exemple, le GET de la figure III.1 est utilisé pour décrire graphiquement le comportement d'un système, ensuite une équation mathématique décrivant le comportement analytique du même système peut être déduite du modèle graphique ainsi obtenu. Dans la majorité des cas, les modèles mathématiques, exprimés dans l'algèbre usuelle, traduisant le comportement d'un GET ne sont pas linéaires, et leur manipulation et résolution restent difficiles. Afin de palier à ce problème, nous exprimons les équations en questions dans une algèbre plus adéquate et facile à exploiter. Nous introduisons pour cela une algèbre plus dédiée à la modélisation et l'analyse des SED. Le modèle GET est donc complété par des équations mathématiques dans l'algèbre des dioïdes pour analyser et évaluer, d'une façon complémentaire et complète, les performances du système modélisé. Quelques éléments de base de cette algèbre seront détaillés dans la section 2.2 et plus de détails sont donnés dans [FGGJJ92].

Définition 2.1.5 *(**Réseaux de Petri avec conflits**). Un réseau de Petri avec conflits est un réseau de Petri qui possède une place avec au moins deux transitions en aval. Un conflit est noté* $[p_i, \{t_1, t_2, ..., t_n\}]$ *avec* $t_1, t_2, ..., t_n$ *étant les transitions de sortie de la place conflictuelle* p_i*.*

Remarque 2.1.3 *Un RdP avec conflits à choix libre est un réseau de Petri dans lequel pour tout conflit* $[p_i, \{t_1, t_2, ..., t_n\}]$ *toutes les transitions* $t_1, t_2, ..., t_n$ *possèdent une et une seule place d'entrée qui est* p_i*. De ce fait, aucune de ces transitions n'est prioritaire pour le franchissement.*

La figure III.2 présente les deux types de RdP avec conflit. La figure III.2a illustre un exemple de RdP à choix libre, où la place P a deux transitions de sortie x_1 et x_2. Si la place P contient un jeton la transition x_1, ou bien x_2, peut exclusivement être franchie. Alors que dans la figure III.2b, la transition x_1 a deux places en amont P_1 et P_2 ce qui élimine le choix libre et donne la priorité de franchissement à la transition x_1. En effet, la transition x_1 est prioritaire dans le cas où P_1 et P_2 contiennent chacune au moins un jeton suite au franchissement respective des transitions u_1 et u_2. La transition x_2 est franchie uniquement dans le cas où P_2 contient un jeton et $M(P_1) = 0$.

Définition 2.1.6 *(**Réseaux de Petri temporisés**). Un réseau de Petri temporisé permet de décrire un système dont le fonctionnement évolue dans le temps. On distingue deux types de réseaux de Petri temporisés, le premier type où les temporisations sont associées aux places, il s'agit dans ce cas de* réseau de Petri P-temporisé*, dans le deuxième type, les temporisations sont associées aux transitions, il s'agit dans ce cas de* réseau de Petri T-temporisé*. Dans notre travail,*



(a) RdP avec choix libre **(b)** RdP sans choix libre

Figure III.2 – Réseaux de Petri avec conflits

nous nous intéressons aux réseaux de Petri P-temporisés. Un réseau de Petri P-temporisé est un doublet $< R, \Theta >$ *tel que :*

- R *est un réseau de Petri marqué,*
- Θ *est une application définie de l'ensemble* P *des places dans l'ensemble des nombres entiers positifs ou nuls.*

La figure III.1, illustrant l'exemple d'un GET, représente un phénomène de synchronisation. La date au plus tôt du $k^{éme}$ franchissement de la transition x_3 est conditionnée par la date du $k^{éme}$ franchissement de la transition x_1 et de la date du $k^{éme}$ franchissement de la transition x_2. De plus, il faut tenir compte du temps de séjour minimal d'un jeton dans les places P_1 et P_2. Nous obtenons donc ces deux inéquations :

$$x_3(k) \geq x_1(k) + 5 \text{ et } x_3(k) \geq x_2(k) + 2$$

Finalement, nous obtenons pour $x_3(k)$, l'inéquation suivante :

$$x_3(k) \geq max(x_1(k) + 5, x_2(k) + 2)$$

$x_3(k)$ se lit "la date du $k^{ième}$ franchissement de la transition x_3".

2.2 Algèbre (max,+)

L'algèbre des dioïdes est apparue comme la structure mathématique adéquate pour modéliser les phénomènes de synchronisation, d'assemblage, de parallélisme. dans le domaine des systèmes à évènement discrets. Cet outil a fait l'objet de nombreuses réalisations pour l'évaluation des performances et pour le contrôle des SED [HOvdW05], [HBL06]. Il est dédié à l'analyse quantitative des propriétés de certains systèmes dont le comportement peut être représenté sous

forme d'équations linéaires. Parmi les algèbres les plus utilisées, nous citons l'algèbre (max,+) et l'algèbre (min,+). Compte tenu de la nature de la problématique que nous abordons dans ce travail, nous nous concentrons sur l'algèbre (max,+) dont nous donnons quelques éléments de base dans ce qui suit. Une représentation plus détaillée se trouve dans [CM89] et [FGGJJ92].

Définition 2.2.1 *(**Dioïde**). Un dioïde $\mathbb{D}$ est un ensemble muni de deux lois internes, notées $\oplus$ et $\otimes$ appelées resp. "addition" et "multiplication", telles que :*
$\forall$ a, b, c $\in \mathbb{D}$, on a :

- *$\oplus$ est associative : $(a \oplus b) \oplus c = a \oplus (b \oplus c)$;*
- *$\oplus$ est commutative : $a \oplus b = b \oplus a$;*
- *$\oplus$ admet un élément neutre, noté ε : $a \oplus \varepsilon = a$;*
- *$\otimes$ est associative : $(a \otimes b) \otimes c = a \otimes (b \otimes c)$;*
- *$\otimes$ admet un élément neutre, noté e : $e \otimes a = a$;*
- *$\otimes$ est distributive par rapport à $\oplus$: $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$;*
- *ε est absorbant pour $\otimes$: $\varepsilon \otimes a = a \otimes \varepsilon = \varepsilon$;*
- *$\oplus$ est idempotent : $a \oplus a = a$.*

De plus, le dioïde $\mathbb{D}$ est dit commutatif si la multiplication l'est. Nous citons par la suite quelques exemples de dioïdes.

- L'ensemble $\mathbb{R} \bigcup \{-\infty\}$ muni des deux opérations *maximum* (notée $\oplus$) et de *l'addition* usuelle (notée $\otimes$) est un dioïde commutatif. Les éléments neutres correspondants aux lois définies sont $\epsilon = -\infty$ pour la loi $\oplus$ (p. ex. $1 \oplus \epsilon = 1$) et $e = 0$ pour la loi $\otimes$ (p. ex. $5 \otimes$ e $= 5$). Ce dioïde est noté $\mathbb{R}_{max}$.
- L'ensemble $\mathbb{R} \bigcup \{+\infty\}$ muni des deux opérations *minimum* (notée $\oplus'$) et de *l'addition* usuelle (notée $\otimes$) est un dioïde commutatif. Les éléments neutres correspondants aux lois définies sont $\epsilon = +\infty$ pour la loi $\oplus'$ (p. ex. $1 \oplus' \epsilon = 1$) et $e = 0$ pour la loi $\otimes$ (p. ex. $5 \otimes$ e $= 5$). Ce dioïde est noté $\mathbb{R}_{min}$.
- Le dioïde matriciel $(\mathbb{Z}^{n \times n}, \oplus, \otimes)$ est l'ensemble des matrices carrées de dimension n à coefficients dans le dioïde des entiers relatifs $\mathbb{Z}$. La somme et le produit de deux matrices ou d'une matrice avec un entier relatif sont définies par :
  pour A, B $\in \mathbb{Z}^{n \times n}$ et $\alpha \in \mathbb{Z}$, on a :

  $(A \oplus B)_{ij} = A_{ij} \oplus B_{ij}$

  $(A \otimes B)_{ij} = \bigoplus_{k=1}^{n} A_{ik} \otimes B_{kj}$

  $\alpha \otimes (A_{ij}) = \alpha \otimes A_{ij}$

Définition 2.2.2 ***Dateur***. *Soit $R = \langle P, T, A, W, M_0 \rangle$ un réseau de Petri. Le dateur associé à chaque transition $x_i \in T$ est noté $x_i(k)$ et représente la date du $k^{ème}$ franchissement de x_i. Ce dateur est défini par l'application suivante :*

$$\begin{aligned} x_i \quad : \quad & \mathbb{N}^* \rightarrow \mathbb{R}_{max} \\ & k \rightarrow x_i(k) \qquad avec : i \in \{1, 2, ..., |T|\} \end{aligned} \tag{III.10}$$

3 Du workflow pattern aux équations (max,+)

L'initiative de transformer les workflow en RdP a été réalisée par Aalst dans [vdA98]. Nous nous basons dans notre travail sur ces transformations, avec plus de détails et de spécifications, en générant plus loin le modèle mathématique dans l'algèbre (max,+) de chaque pattern. Nous n'allons pas parcourir les quarante trois patterns définis dans le travail de Aalst [vdA98], mais nous allons nous concentrer sur ceux qui seront utilisés pour la composition de services dans le scénario que nous définissons dans le chapitre 5, §4.2. Ce travail a été motivé par le fait que ces modèles sont particulièrement utiles dans le contexte de la composition de services Web et qu'ils sont souvent soutenus par des langages de composition tels que WS-BPEL.

Dans les modèles RdPs modélisés ci-dessous, les places représentent les tâches ou les activités dans le processus tandis que les transitions modélisent les états (début et fin) de chaque activité ; elles décrivent quand une activité commence et quand elle se termine.

3.1 Le pattern séquence

Le workflow pattern séquence est le pattern le plus fondamental et le plus simple pour la construction d'un block de processus. Il est utilisé pour construire une série de tâches consécutives qui s'exécute l'une après l'autre. Une activité du workflow est déclenchée une fois que la précédente est terminée. La figure III.3 illustre le modèle RdP associé à ce pattern. Deux tâches font partie d'une même séquence s'il y a un arc liant l'une à l'autre. Un exemple d'utilisation de ce pattern est le distributeur de billet, où la tâche de vérification du compte s'exécute une fois que les tâches d'authentification et de lecture de données de la carte bancaire soient terminées.

Nous rappelons que les temporisations τ_j représentent le temps nécessaire pour qu'une tâche soit finie. De même, nous rappelons que chaque temporisation correspond au meilleur temps retenu par l'exécution d'une tâche. Autrement dit, une temporisation correspond au temps

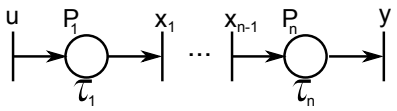


Figure III.3 – Le workflow pattern séquence

d'exécution d'une tâche par le service qui répond le mieux aux critères de la qualité de service comme expliqué dans la section 1.3.1. Les différentes temporisations sont donc obtenues après la phase de sélection de services.

Dans la figure III.3, u (resp. y et x_i) représente la transition d'entrée (resp. de sortie et interne) du modèle RdP.

Le comportement analytique du pattern est représenté par le système (III.11).

$\forall k \geq 1$:

$$\begin{cases} x_1(k) &= \tau_1 \otimes u(k) \\ x_2(k) &= \tau_2 \otimes x_1(k) \\ ... &= ... \\ x_{n-1}(k) &= \tau_{n-1} \otimes x_{n-2}(k) \\ y(k) &= \tau_n \otimes x_{n-1}(k) \end{cases} \tag{III.11}$$

Le système (III.11) s'écrit sous la forme matricielle par les équations suivantes :

$\forall k \geq 1$:

$$\begin{cases} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \end{pmatrix}(k) = \begin{pmatrix} \epsilon & & \cdots & & \epsilon \\ \tau_2 & \epsilon & \cdots & & \epsilon \\ \epsilon & \tau_3 & & & \\ \vdots & & \ddots & \ddots & \vdots \\ \epsilon & \cdots & \epsilon & \tau_{n-1} & \epsilon \end{pmatrix} \otimes \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \end{pmatrix}(k) \oplus \begin{pmatrix} \tau_1 \\ \epsilon \\ \vdots \\ \\ \epsilon \end{pmatrix} \otimes u(k) \\ y(k) = \begin{pmatrix} \epsilon & \cdots & \epsilon & \tau_n \end{pmatrix} \otimes \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{pmatrix}(k) \end{cases} \tag{III.12}$$

Le système (III.12) s'écrit sous la forme du système (III.13).

$$\begin{cases} X(k) &= A_{seq} \otimes X(k) \oplus B_{seq} \otimes U(k) \\ Y(k) &= C_{seq} \otimes X(k) \end{cases} \tag{III.13}$$

sachant que $X(k)$ est le vecteur d'état du modèle ayant comme composantes $x_1(k)$, $x_2(k)$,

..., $x_{n-1}(k)$. $U(k)$ est le vecteur d'entrée, et $Y(k)$ constitue le vecteur de sortie. Pour ce cas, les vecteurs U et Y sont composés chacun d'une seule composante, nous notons donc $Y(k) = y(k)$ et $U(k) = u(k)$. $A_{seq} \in \mathcal{M}^{(n-1)\times(n-1)}(\mathbb{R}_{max})$, $B_{seq} \in \mathcal{M}^{(n-1)\times m}(\mathbb{R}_{max})$ et $C_{seq} \in \mathcal{M}^{q\times(n-1)}(\mathbb{R}_{max})$ sont des matrices caractéristiques du modèle dont les composantes représentent les données du système. (n-1) (resp. m et q) est le nombre de transitions internes (resp. d'entrée et de sortie, dans notre cas $m = q = 1$) du modèle.

3.2 Le pattern synchroniseur

Ce pattern décrit le phénomène de synchronisation. Il est définit comme étant la convergence d'au moins deux ou plusieurs branches en une seule branche postérieure, de telle sorte que cette dernière ne peut être activée que si toutes les branches antérieures le sont. Par exemple l'activité d'expédition de la marchandise s'exécute immédiatement après l'achèvement des différentes activités antérieures telles que : disponibilité de la marchandise, validation de la commande, validation du paiement, etc.

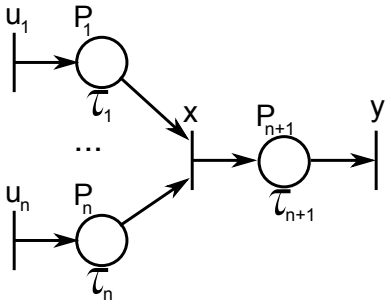


Figure III.4 – Le workflow pattern synchroniseur

Le comportement graphique du pattern est décrit par le modèle de la figure III.4. La transition x ne peut être franchie que si toutes les transitions $u_1, ..., u_n$ le sont. La contrainte de blocage peut apparaître dans ce pattern. En effet, si la tâche représentée par une des branches entrantes n'est pas finie, la synchronisation ne pourra s'activer et sera dans un état d'attente de la réponse provenant de la branche en question.

L'expression donc d'une date de franchissement de la transition x s'écrit : $x(k) = max\{\tau_1 + u_1(k), ..., \tau_n + u_n(k)\}$. Par conséquent, le comportement analytique de ce pattern dans l'algèbre (max,+) est exprimé par le système d'équations (III.14).

$$\begin{cases} x(k) & = \tau_1 \otimes u_1(k) \oplus \tau_2 \otimes u_2(k) \oplus ... \oplus \tau_n \otimes u_n(k) \\ y(k) & = \tau_{n+1} \otimes x(k) \end{cases} \tag{III.14}$$

La première équation s'écrit d'une façon plus générale sous la forme de l'équation (III.15).

$$x(k) = \bigoplus_{i=1}^{n} \tau_i \otimes u_i(k) \tag{III.15}$$

Le système d'équations (III.14) s'écrit sous la forme matricielle suivante :

$$\begin{cases} x(k) & = \begin{pmatrix} \tau_1 & \tau_2 & \cdots & \tau_n \end{pmatrix} \otimes \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix} (k) \\ y(k) & = \tau_{n+1} \otimes x(k) \end{cases} \tag{III.16}$$

Le système (III.16) s'écrit sous la forme du système d'équations (III.17).

$$\begin{cases} X(k) & = B_{sync} \otimes U(k) \\ Y(k) & = C_{sync} \otimes X(k) \end{cases} \tag{III.17}$$

où $B_{sync} \in \mathcal{M}^{1\times n}(\mathbb{R}_{max})$ et $C_{sync} \in \mathcal{M}^{1\times 1}(\mathbb{R}_{max})$ sont des matrices caractéristiques contenant les données du pattern.

3.3 Le pattern choix différé

La figure III.5 représente le pattern *choix différé*. Lorsque la place P contient un jeton, toutes les transitions sont en situation de conflit. Il s'agit donc d'un RdP à choix libre. Ainsi, la transition qui sera franchie peut être choisie d'une façon aléatoire. Cette politique ne peut répondre aux besoins de l'utilisateur d'une façon efficace que lorsque certains critères de choix sont imposés. L'objectif est d'analyser et évaluer les performances des systèmes avec conflits, en intégrant les contraintes de choix de franchissement des transitions en situation de conflits dans les équations mathématiques modélisant de tels systèmes. Pour ce faire, nous commençons par définir quelques critères de choix. Un choix peut être fait, par exemple, en fonction de la disponibilité des services ou des ressources. Le service le plus approprié pour répondre à une requête, ou réaliser une tâche dans un scénario, sera choisi par le franchissement de la transition qui le représente dans le modèle RdP. Par exemple, dans le contexte de l'interaction de services, le choix est fait selon les critères de disponibilité. Il peut également se faire en fonction du service offrant une prestation optimale (coût, temps de réponse, fiabilité, etc.).

Dans le but de gérer le conflit qui apparaît dans le modèle de la figure III.5, nous devons définir une politique de routage permettant de choisir la manière dont les transitions en conflit

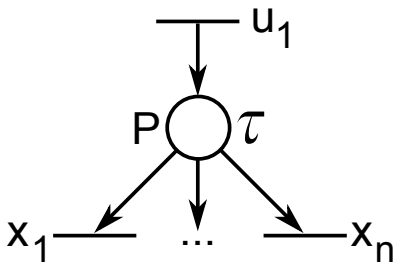


Figure III.5 – Le workflow pattern choix différé

peuvent être franchies, et aussi dans quel ordre et selon quels critères. Par ailleurs afin de modéliser ce système par des équations (max,+), nous avons introduit la notion de *jeton virtuel* [cbsmW10].

Définition 3.3.1 ***(Jeton virtuel)***. *L'unique jeton dans une place conflictuelle participe au franchissement réel d'une seule transition de sortie, les autres transitions, en situation de conflit avec la seule transition choisie pour le franchissement, ne sont pas réellement franchies, mais on suppose qu'elles le sont virtuellement par des jetons virtuels. L'introduction de ces tirs virtuels facilite la description du comportement du système par des représentations linéaires dans l'algèbre (max,+).*

Afin de représenter ces franchissements virtuels d'une façon formelle, nous définissons la fonction suivante :

$$\begin{aligned} f \quad : \quad & T \rightarrow \{\epsilon, e\} \\ & x_i \rightarrow f(x_i) \qquad avec : i \in \{1, 2, ..., |T|\} \end{aligned} \tag{III.18}$$

où $|T|$ est le cardinal de l'ensemble des transitions T. Pour simplifier l'équation (III.18), nous notons f_{x_i} au lieu de $f(x_i)$ dans ce qui suit.

Formellement, soit $\mathbf{P_c} \subset \mathbf{P}$ l'ensemble des places telles que pour toute place $P_l \in \mathbf{P_c}$, on a $|P_l^\circ| > 1$, nous appelons aussi P_c l'ensemble des places conflictuelles. Pout toute place $P_l \in \mathbf{P_c}$ et pour $k \in \mathbb{N}^*$ on a :

$$\begin{cases} \forall x_i \in P_l^\circ, \exists!\ x_j \in P_l^\circ; & x_j(k) = \alpha \otimes f_{x_j}(k) = \alpha \quad (\alpha \in \mathbb{R}^+) \\ \forall x_i \in P_l^\circ \backslash \{x_j\}; & x_i(k) = \alpha \otimes f_{x_i}(k) = \epsilon \end{cases}$$

où x_j est franchie réellement pour la $k^{ème}$ fois (c.-à-d. $f_{x_j} = e$), toutes les autres transitions x_i (avec i différent de j) sont franchies virtuellement pour la $k^{ème}$ fois (c.-à-d. $f_{x_i} = \epsilon$). Sachant

que l'application dateur est une application monotone, ce franchissement virtuel vérifie cette propriété même si la place P_l contient plus d'un jeton.

Si maintenant, nous souhaitons définir un ordre de franchissement des différentes transitions $x_i \in P_l^\circ$ (avec $P_l^\circ \in \mathbf{P_c}$), nous devons définir une fonction de routage et appliquer la stratégie du franchissement virtuel définie ci-dessus. En effet, dans la figure III.5, nous avons décidé de franchir les transitions en aval de la place P selon un ordre prioritaire. Nous considérons donc les franchissements des différentes transitions x_i de la façon suivante : le premier franchissement (k = 1) (resp. le deuxième (k = 2), ..., le $n^{ème}$ (k = n)) de la transition u_1 est suivi du premier franchissement de x_1 (resp. x_2, ..., x_n), le $(n+1)^{ème}$ franchissement de u_1 est suivi du $2^{ème}$ franchissement de x_1 et ainsi de suite. Cet ordonnancement nous permet d'arbitrer le conflit en évitant le franchissement aléatoire des transitions. Suite à ce choix de franchissement de transitions en situation de conflit, nous nous intéressons à la génération des équations (max,+) en tenant compte de ces conflits et des franchissements réels et virtuels. Formellement cet ordre de franchissement des transitions est exprimé par la fonction de routage définie par (III.19).

Définition 3.3.2 ***(Fonction de routage)***
$\forall\ P_l \in \mathbf{P_c}$, $\forall\ x_i \in P_l{}^\circ$ *avec* $1 \leq i \leq |P_l{}^\circ|$, $\forall k \geq 1$ *;*

$$f_{x_i}(k) = \begin{cases} e & si\ i \equiv k\ mod(|P_l{}^\circ|) \\ \epsilon & sinon \end{cases} \tag{III.19}$$

avec $|P_l{}^\circ|$ *le cardinal de l'ensemble* $P_l{}^\circ$*.*

Pour le pattern de la figure III.5, nous avons une seule place conflictuelle P, avec :
- $|P^\circ| = n$
- $T_{Conf} = P^\circ = \{x_1, x_2, ..., x_n\}$

Le comportement analytique de ce pattern est décrit comme suit :$\forall k \geq 1$:

$$\begin{cases} x_1(k) & = \tau \otimes u_1(k) \otimes f_{x_1}(k) \\ x_2(k) & = \tau \otimes u_1(k) \otimes f_{x_2}(k) \\ \quad ... & = ... \\ x_n(k) & = \tau \otimes u_1(k) \otimes f_{x_n}(k) \end{cases} \tag{III.20}$$

Le système (III.20) peut être écrit sous la forme matricielle :

$$X(k) = F(k) \otimes B \otimes U(k) \tag{III.21}$$

D'une façon générale, le système (III.21) peut s'exprimer par les équations du système (III.22) pour un modèle complet où les variables d'entrée, les variables d'état et les variables de sortie sont définies. Ainsi, le modèle (max,+) générique est le suivant :

$$\begin{cases} X(k) &= A \otimes F_1(k) \otimes X(k) \oplus F_2(k) \otimes B \otimes U(k) \\ Y(k) &= C \otimes F_3(k) \otimes X(k) \end{cases} \tag{III.22}$$

avec $U(k)$ le vecteur d'entrée, $X(k)$ le vecteur d'état, $Y(k)$ le vecteur de sortie. A $\in \mathcal{M}_{n\times n}(\mathbb{R}_{max})$, B $\in \mathcal{M}_{n\times m}(\mathbb{R}_{max})$ et C $\in \mathcal{M}_{q\times n}(\mathbb{R}_{max})$ sont des matrices caractéristiques. n (resp. m et q) est le nombre des transitions internes (resp. d'entrée et de sortie) du modèle. Les matrices de routage $F_1(k) \in \mathcal{M}_{n\times n}(\mathbb{R}_{max})$, $F_2(k) \in \mathcal{M}_{n\times n}(\mathbb{R}_{max})$ et $F_3(k) \in \mathcal{M}_{n\times n}(\mathbb{R}_{max})$. Les matrices A, B, C sont exprimées en fonction des données du système. Les valeurs des éléments des matrices $F_{1,2,3}$ varient en fonction de k. Ces matrices permettent de déterminer les transitions en situation de conflits qui seront réellement franchies pour tout évènement k.

Pour l'exemple de la figure III.5 dont l'équation est exprimée par (III.21), la matrice de routage F(k) est donnée par :

$$F(k) = \begin{pmatrix} f_{x_1}(k) & \epsilon & \cdots & \epsilon \\ \epsilon & \ddots & & \vdots \\ \vdots & & \ddots & \epsilon \\ \epsilon & \cdots & \epsilon & f_{x_n}(k) \end{pmatrix} \tag{III.23}$$

3.4 Le pattern multi-fusion

Ce pattern est définit par la fusion de deux ou plusieurs branches non-synchronisées en une seule branche postérieure, comme illustré dans la figure III.6. Chaque activation d'une branche entrante dans le processus participe, indépendamment des autres branches entrantes, à l'activation de la branche postérieure. Le pattern multi-fusion est un moyen de fusionner les branches distinctes en une seule branche. Bien que plusieurs chemins d'exécution sont fusionnés, il n'y a pas de synchronisation des flux de contrôle, chaque branche active va donc joindre la branche résultante fusionnée.

Le fonctionnement de ce modèle est illustré dans la figure III.6. Lors de l'exécution de l'une des transitions $x_1, x_2, ...,$ ou x_n, la transition de sortie y est validée et peut être franchie après un certain temps. Cependant, il n'est pas évident d'exprimer formellement le nombre de franchissement de la transition y en fonction du nombre de franchissements de ses transitions en amont. Pour pouvoir décrire ce mode de fonctionnement par des équations (max, +) et

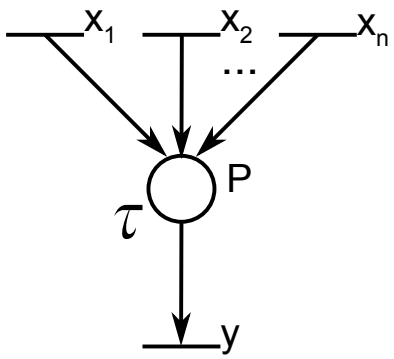


Figure III.6 – Le workflow pattern multi-fusion

afin de faciliter son analyse mathématique, nous avons défini des places *compteurs* que nous associons aux transitions modélisant les branches parallèles entrantes. Ainsi, nous associons une place compteur à chacune des transitions x_1, x_2,..., et x_n. Ces places compteurs sont des places puits $P_1, P_2, ..., etP_n$ comme illustré dans la figure III.7. Nous signalons que l'ajout de ces places compteurs n'aura pas d'influence sur la vérification de certaines propriétés du modèle RdP, notamment la bornitude. En effet, chacune de ces places, dont la capacité est supposée infinie, sert uniquement à renseigner le nombre de franchissement de la transition en amont. Elle ne participe en aucun cas au changement de la dynamique du modèle RdP.

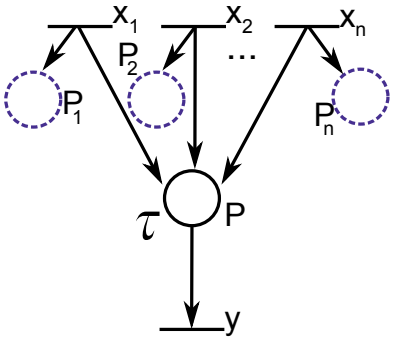


Figure III.7 – Le workflow pattern multi-fusion avec les places compteurs

En utilisant ces places compteurs, le comportement du pattern modélisé par le modèle RdP de la figure III.7 est représenté par les équations du système (III.24). Ces équations donneront une expression explicite de différentes dates de franchissement de la transition y en fonction de celles en amont. Pour illustrer cette approche, nous commençons par un exemple de franchissement des transitions x_1, x_2 et x_3 (on considère n = 3 le nombre de transitions en amont de la place P de la figure III.7).

1. Pour k=1 : le premier franchissement de la transition y correspond au premier franchissement de la transition x_1.

2. Pour k=2 : le deuxième franchissement de la transition y correspond au deuxième franchissement de la transition x_1.
3. Pour k=3 : le troisième franchissement de la transition y correspond au premier franchissement de la transition x_2.

Les équations mathématiques correspondantes à cet ordre de franchissement sont décrites ci-après :

$$\begin{cases} y(1) & = \tau \otimes x_1(1 - (0 \otimes 0)) \oplus \tau \otimes x_2(1 - (1 \otimes 0)) \oplus \tau \otimes x_3(1 - (1 \otimes 0)) \\ & = \tau \otimes x_1(1) \oplus \epsilon \oplus \epsilon \\ y(2) & = \tau \otimes x_1(2 - (0 \otimes 0)) \oplus \tau \otimes x_2(2 - (2 \otimes 0)) \oplus \tau \otimes x_3(2 - (2 \otimes 0)) \\ & = \tau \otimes x_1(2) \oplus \epsilon \oplus \epsilon \\ y(3) & = \tau \otimes x_1(3 - (1 \otimes 0)) \oplus \tau \otimes x_2(3 - (2 \otimes 0)) \oplus \tau \otimes x_3(3 - (2 \otimes 1)) \\ & = \tau \otimes x_1(2) \oplus \tau \otimes x_2(1) \oplus \epsilon = \tau \otimes x_2(1) \end{cases} \tag{III.24}$$

D'une façon plus générale, le comportement analytique du modèle de la figure III.7, peut s'écrire de la façon suivante :

$$y(k) = \bigoplus_{i=1}^{n} \tau \otimes x_i(k - \bigotimes_{j=1, j\neq i}^{n} m_{P_j}) \tag{III.25}$$

Avec n est le nombre de transitions en amont de la place P, k est le nombre de franchissement de y, et finalement m_{p_j} est le marquage de la place puit p_j juste avant le $k^{ème}$ franchissement de la transition y. Ce raisonnement suppose que les dates des franchissements des transitons en amont de la place P (c.-à-d. x_1, x_2, ...x_n) sont différentes. Dans le cas contraire, nous observons une légère modification au niveau des équations (max,+).

L'écriture matricielle de l'équation (III.25) est donnée comme suit :

$$\begin{cases} Y(k) & = \tau \otimes x_1(k - \bigotimes_{j=2, j\neq 1}^{n} m_{P_j}) \oplus \tau \otimes x_2(k - \bigotimes_{j=1, j\neq 2}^{n} m_{P_j}) \oplus ... \oplus \tau \otimes x_n(k - \bigotimes_{j=1, j\neq n}^{n-1} m_{P_j}) \\ & = \tau \otimes x_1(k - k_1) \oplus \tau \otimes x_2(k - k_2) \oplus ... \oplus \tau \otimes x_n(k - k_n) \end{cases} \tag{III.26}$$

Avec : $k_i = \bigotimes_{j=1, j\neq i}^{n} m_{P_j}$ pour tout i, $1 \leq i \leq n$,

Sachant que $k_i \leq k$ pour tout i, on en déduit la forme matricielle suivante :

$$y(k) = A_{k_1} \otimes x_1(k - k_1) \oplus A_{k_2} \otimes x_2(k - k_2) \oplus ... \oplus A_{k_n} \otimes x_n(k - k_n) \tag{III.27}$$

L'équation (III.27) devient :

$$Y(k) = \bigoplus_{i=1}^{n} A_{k_i} \otimes X(k - k_i) \tag{III.28}$$

Où $A_{k_i} \in \mathbb{R}_{max}^{1\times n}$ avec $A_{k_i} = \begin{pmatrix} \epsilon & \cdots & \epsilon & \tau & \epsilon & \cdots & \epsilon \end{pmatrix}$

L'emplacement de τ correspond à la $i^{éme}$ position dans le vecteur A_{k_i}.

et $X(k - k_i) = \begin{pmatrix} x_1(k - k_1) \\ x_2(k - k_2) \\ \vdots \\ x_n(k - k_n) \end{pmatrix}$

3.5 Le pattern fractionnement parallèle

Le pattern fractionnement parallèle décrit la divergence d'une branche en deux ou plusieurs branches parallèles dont chacune va être exécutée simultanément. Ce pattern peut se voir comme le synonyme de "AND-split". Par exemple, lorsqu'un client paye ses achats, ses marchandises sont emballées et sa facture est imprimée. En d'autres termes, le pattern fractionnement parallèle permet à un processus d'être divisé en deux ou plusieurs processus qui peuvent être exécutés simultanément. La figure III.8 illustre le modèle RdP de ce pattern. Après l'exécution de la transition x_1, chacune des transitions x_2, ...,x_n est validée et peut être exécutée après un temps donné.

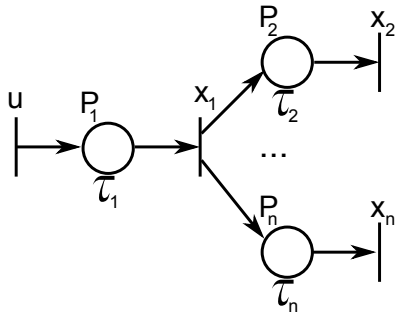


Figure III.8 – Le workflow pattern fractionnement parallèle

Ce pattern correspond au phénomène de parallélisme tel qu'il est connu dans les RdPs, son comportement analytique est décrit par le système d'équations (III.29).

$\forall k \geq 1$:

$$\begin{cases} x_1(k) & = \tau_1 \otimes u(k) \\ x_2(k) & = \tau_2 \otimes x_1(k) \\ \quad ... & = ... \\ x_n(k) & = \tau_n \otimes x_1(k) \end{cases} \tag{III.29}$$

L'écriture matricielle du système (III.29) est la suivante :
$\forall k \geq 1$:

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}(k) = \begin{pmatrix} \epsilon & \cdots & \epsilon \\ \tau_2 \ \epsilon & \cdots & \epsilon \\ \vdots & \ddots & \vdots \\ \tau_n \ \epsilon & \cdots & \epsilon \end{pmatrix} \otimes \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}(k) \oplus \begin{pmatrix} \tau_1 \\ \epsilon \\ \vdots \\ \epsilon \end{pmatrix} \otimes u(k) \tag{III.30}$$

Plus généralement, cette équation devient :

$$X(k) = A \otimes X(k) \oplus B \otimes U(k) \tag{III.31}$$

Avec A $\in \mathcal{M}_{n\times n}(\mathbb{R}_{max})$ et B $\in \mathcal{M}_{n\times m}(\mathbb{R}_{max})$ sont les matrices caractéristiques, n (resp. m) est le nombre des transitions internes (resp. d'entrée).

3.6 Le pattern choix conditionnel

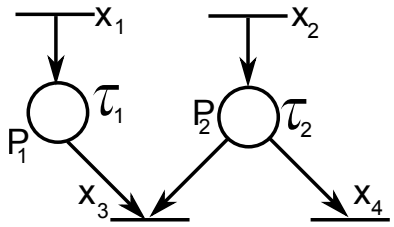


Figure III.9 – Le workflow pattern choix conditionnel

Le modèle RdP avec conflit de la figure III.9 représente le pattern *choix conditionnel*. La résolution du conflit revient donc à choisir la transition à franchir parmi les transitions en situation de conflit. D'un point de vue structurel, les transitions en situation de conflit sont validées en fonction de la disponibilité des jetons dans leurs places en amont. En effet, pour les transitions x_3 et x_4, où $x_3 \in P_1{}^\circ \bigcap P_2{}^\circ$ et $x_4 \in P_2{}^\circ$, les règles de franchissement sont données comme suit :

1. Si chacune des deux places P_1 et P_2 contient un jeton, et les temps de séjour τ_1 et τ_2 sont achevés, alors la priorité de franchissement est donnée à la transition x_3 ;
2. Si la place P_1 ne contient aucun jeton, tandis qu'un jeton se trouve dans la place P_2 et le temps de séjour τ_2 est achevé, alors x_4 sera franchie ;
3. Si la place P_1 contient un jeton et P_2 ne contient aucun jeton, alors aucune de ces deux transitions n'est validée. Dans ce cas, pour franchir x_3 ou x_4, il faut attendre l'arrivée d'un autre jeton dans la place P_2.

Dans le but de décrire le comportement de ce pattern d'une façon formelle par des équations (max, +), nous réutilisons le franchissement virtuel, des transitions en situation de conflit, définit dans la section 3.3. En effet, pour chaque franchissement de la transition x_2 du modèle de la figure III.9 et après l'achèvement du temps τ_2, une des deux transitions x_3 ou x_4 est franchie réellement, alors que l'autre est supposée être franchie virtuellement. Pour exprimer ces modes de franchissement, nous définissons les deux fonctions Ψ et $\overline{\Psi}$ par :

$$\Psi_{a\leq b} = \begin{cases} e & si\ a \leq b \\ \epsilon & sinon \end{cases} \tag{III.32}$$

et

$$\overline{\Psi}_{a\leq b} = \begin{cases} \epsilon & si\ a \leq b \\ e & sinon \end{cases} \tag{III.33}$$

Nous introduisons également le marquage de certaines places du modèle RdP dans les équations. Ainsi $m_P(x_i(k))$ signifie le marquage de la place P à l'instant $x_i(k)$. Nous supposons que $\forall k \leq 0$, $x_i(k) = \epsilon$ et le marquage d'une place à cet instant est nul.

Afin de déterminer les différentes dates des franchissements réels de la transition x_3 (resp. x_4), nous supprimons les valeurs ϵ (correspondant aux franchissements virtuels) de l'ensemble des résultats obtenus, puis nous décalons les valeurs des dates de franchissements réels qui prennent les places des dates des franchissements virtuels. Ainsi, nous remarquons que le $k^{ème}$ franchissement des transitions x_3 et x_4 ne correspondent pas, en général, au $k^{ème}$ franchissement des transitions x_1 et x_2. Ces opérations de mise en forme des résultats obtenus sont illustrées dans l'exemple suivant.

Le tableau III.1 exprime les valeurs des temporisations associées au modèle RdP de la figure III.9.

Nous considérons dans un premier temps les franchissements des transitions x_1, x_2, x_3 et x_4 en tenant compte des franchissements virtuels. Les dates de ces franchissements sont

Tableau III.1 – Les valeurs numériques des temporisations du modèle RdP

Temporisation	Valeur
τ_1	6
τ_2	1

représentées dans le tableau III.2.

Tableau III.2 – Dates des franchissements réels et virtuels des transitions

k	x_1	x_2	x_3	x_4
1	10	11	ϵ	12
2	20	22	23	ϵ
3	30	33	34	ϵ
4	40	44	45	ϵ
5	50	55	56	ϵ
6	60	66	67	ϵ

Le tableau III.3 présente les dates de franchissement des transitions du modèle RdP après la mise en forme des résultats donnés dans le tableau III.2 après la résolution du modèle (max,+) décrivant le comportement de la figure III.9.

Tableau III.3 – Dates des franchissements réels des transitions

k	x_1	x_2	x_3	x_4
1	10	11	23	12
2	20	22	34	-
3	30	33	45	-
4	40	44	56	-
5	50	55	67	-
6	60	66	-	-

Les équations (max,+) décrivant les dates de franchissement (réelles et virtuelles) des transitions x_3 ou x_4, en fonction de celles des transitons x_1 ou x_2, sont donc données par :

$\forall k \geq 1$:

$$\begin{cases} x_3(k) & = [\tau_1 \otimes x_1(k) \oplus \tau_2 \otimes x_2(k)] \otimes \overline{\Psi}_{\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k)} \\ & \quad \oplus [\tau_1 \otimes x_1(k - m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1)) \oplus \tau_2 \otimes x_2(k)] \otimes \Psi_{m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1) \neq 0} \\ x_4(k) & = \tau_2 \otimes x_2(k) \otimes [\Psi_{\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k) \wedge (m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1)) = 0}] \end{cases} \quad \text{(III.34)}$$

L'opérateur "∧" est défini comme "et", $a \wedge b$ signifie que les deux conditions a et b doivent être vérifiées simultanément.

Concrètement, les termes de l'équation exprimant les dates de franchissement de la transition x_3 sont donnés par :

- $[\tau_1 \otimes x_1(k) \oplus \tau_2 \otimes x_2(k)] \otimes \overline{\Psi}_{\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k)} = \tau_2 \otimes x_2(k) \neq \epsilon$, si $\tau_1 \otimes x_1(k) \leq \tau_2 \otimes x_2(k)$: ceci signifie que le franchissement réel de la transition x_3 se produit quand un jeton arrive en premier à la place P_1 ensuite un autre jeton arrive dans la place P_2 et leurs temps de séjour dans ces places sont achevés. Il s'agit, dans ce cas, d'un phénomène de synchronisation. Si, par contre, $\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k)$, le premier terme de l'équation s'annule ($= \epsilon$) par la fonction $\overline{\Psi}$.
- $[\tau_1 \otimes x_1(k - m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1)) \oplus \tau_2 \otimes x_2(k)] \otimes \Psi_{m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1) \neq 0}$: ce terme s'annule quand le marquage de la place P_1 (m_{P_1}) à l'instant $(\tau_2 \otimes x_2(k) - \tau_1)$ est nul. Dans le cas contraire, si le marquage n'est pas nul à cet instant, le franchissement de la transition x_3 se produit quand un jeton arrive dans la place P_2. Nous introduisons dans le terme $\tau_1 \otimes x_1(k - m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1))$ le nombre de jetons $k - m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1)$ pour tenir compte de tous les jetons arrivés entre temps dans la place P_2 jusqu'à l'instant $(\tau_2 \otimes x_2(k) - \tau_1)$ et non uniquement le dernier jeton arrivé avant cet instant.

Les termes de l'équation exprimant les dates de franchissement de la transition x_4 sont donnés par :

- $x_4(k) = \tau_2 \otimes x_2(k)$ si les deux conditions suivantes sont vérifiées : i) $\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k)$ ce qui signifie qu'un jeton arrive à la place P_2 et y expire son temps de séjour avant l'arrivée d'un autre jeton dans la place P_1 ; ii) $m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1) = 0$, c'est-à-dire que le marquage de P_1 à l'instant $(\tau_2 \otimes x_2(k) - \tau_1)$ est nul. Dans ce cas la priorité de franchissement est donnée à la transition x_4.

Dans le cas où la transition x_3 (resp. x_4) est franchie réellement pour la $k^{ème}$ fois, x_4 (resp. x_3) est franchie virtuellement pour la $k^{ème}$ fois.

L'écriture matricielle des équations (III.34) est donnée par :

$$\left\{ \; X(k) \;\; = A_0(k) \otimes X(k) \oplus A_1(k) \otimes X(k - k_1) \right. \tag{III.35}$$

Les matrices caractéristiques $A_0(k)$ et $A_1(k)$ sont données par :

$$\mathbf{A_0(k)} = \begin{pmatrix} e & \epsilon & \epsilon & \epsilon \\ \epsilon & e & \epsilon & \epsilon \\ A_{31}(k) & A_{32}(k) & \epsilon & \epsilon \\ \epsilon & A_{42}(k) & \epsilon & \epsilon \end{pmatrix} \tag{III.36}$$

$$\mathbf{A_1(k)} = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \\ \tilde{A}_{31}(k) & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix} \tag{III.37}$$

Avec :

$$\begin{cases} A_{31}(k) & = \tau_1 \otimes \overline{\Psi}_{\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k)} \\ A_{32}(k) & = \tau_2 \otimes [\overline{\Psi}_{\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k)} \oplus \Psi_{m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1) \neq 0}] \\ A_{42}(k) & = \tau_2 \otimes \Psi_{\tau_1 \otimes x_1(k) > \tau_2 \otimes x_2(k) \wedge (m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1) = 0)} \\ \tilde{A}_{31}(k) & = \tau_1 \otimes \Psi_{m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1) \neq 0} \\ k_1 & = m_{P_1}(\tau_2 \otimes x_2(k) - \tau_1) \end{cases}$$

Il s'agit ici d'un système matriciel non stationnaire dont les matrices caractéristiques dépendent du paramètre k. Au même titre que les systèmes (max, +) donnés auparavant et dans lesquels les matrices de routage apparaissent, cet aspect de non stationnarité est dû aux conflits figurés dans les modèles RdP associés.

Le système (III.35) peut être généralisé, pour un modèle complet, par :

$$\begin{cases} X(k) & = A_0(k) \otimes X(k) \oplus A_1(k) \otimes X(k - k_i) \oplus B(k) \otimes U(k) \\ Y(k) & = C(k) \otimes X(k) \end{cases} \tag{III.38}$$

Avec $U(k)$ le vecteur d'entrée, $X(k)$ le vecteur d'état, $Y(k)$ le vecteur de sortie et $A(k)$, $B(k)$ et $C(k)$ sont les matrices non stationnaires du système.

3.7 Exemple de composition de patterns

Nous allons maintenant étudier un exemple d'application de composition d'un ensemble de patterns. L'objectif est de regrouper plusieurs patterns en un modèle graphique représentant le comportement d'un scénario donné. A partir de ce modèle, nous allons développer les équations mathématiques associées, afin d'analyse et d'évaluer les performances du scénario. Nous consi-

dérons le modèle de la figure III.10 comme exemple d'application de la composition de patterns. Ce modèle représente la composition de trois services S_1, S_2 et S_3. L'interaction entre les services est décrite comme suit : lors du franchissement de la transition u, la place P_1 contiendra un jeton modélisant l'arrivée d'une requête. Pour répondre à cette requête, trois services sont candidats, mais seulement un et un seul service sera sélectionné pour effectuer cette tâche. Le choix d'un service parmi les trois candidats conduit à une situation de conflit au niveau de la place P_1. Cela signifie que les trois transitions x_1,x_2 et x_3 sont activées, mais une seule peut être franchie par le seul jeton présent dans la place P_1. Les franchissements des transitions x_4, x_5 et x_6 représentent resp. la fin de l'exécution des services S_1, S_2 et S_3. La réponse de la requête, fournie par un des trois services est représentée par le jeton ajouté dans la place P_5 et qui est ensuite envoyé au demandeur *via* le franchissement de la transition y. Comme nous pouvons le remarquer, ce modèle est composé de trois patterns à savoir, le *pattern séquence*, le *pattern choix différé* et le *pattern multi-fusion*. Le pattern à choix différé est entouré par des pointillés (partie supérieure du modèle de la figure III.10), le pattern multi-fusion est entouré par une ligne grise (partie inférieure du modèle de la figure III.10). Les modèles entourés par des pointillés gris (notés par S_1, S_2 et S_3) représentent le début et la fin des services.

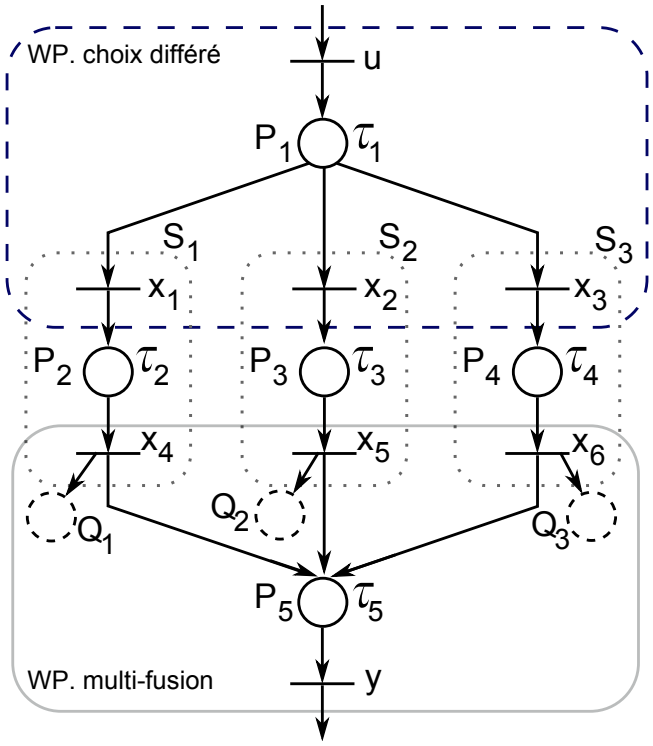


Figure III.10 – Exemple de composition de patterns

Dans un premier temps, nous allons définir les équations (max,+) de chacun de ces pat-

terns séparément. Ensuite, nous procédons à rassembler ces équations en une seule équation représentant le comportement du modèle global.

Le comportement du pattern à choix différé est décrit par l'équation (III.39) comme défini dans la section 3.3.

$$X_c(k) = F(k) \otimes B_c \otimes U(k) \tag{III.39}$$

Avec $X_c(k) \in \mathbb{R}_{max}^{3\times1}$, $F(k) \in \mathbb{R}_{max}^{3\times3}$, $B_c \in \mathbb{R}_{max}^{3\times1}$ et $U(k) = u(k)$.

L'équation (III.40) décrit le comportement du pattern multi-fusion comme expliqué dans la section 3.4.

$$Y(k) = \bigoplus_{i=1}^{n} A_{k_i} \otimes X_f(k - k_i) \tag{III.40}$$

Avec : $X_f(k) \in \mathbb{R}_{max}^{3\times1}$, $A_{k_i} \in \mathbb{R}_{max}^{1\times3}$ et $Y(k) = y(k)$.

Le système d'équations (III.41) définit le comportement du modèle global illustré dans la figure III.10

$$\begin{cases} X(k) & = A \otimes F_1(k) \otimes X(k) \oplus F_2(k) \otimes B \otimes U(k) \\ Y(k) & = \bigoplus_{i=1}^{n} C_{k_i} \otimes X(k - k_i) \end{cases} \tag{III.41}$$

Avec :

– $X \in \mathcal{M}_{6\times1}(\mathbb{R}_{max})$, où $X^t(k) = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{pmatrix}(k)$;

– $A \in \mathcal{M}_{6\times6}(\mathbb{R}_{max})$, où $A = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \vdots \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \vdots \\ \tau_2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \tau_3 & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \tau_4 & \epsilon & \epsilon & \epsilon \end{pmatrix}$;

– $B \in \mathcal{M}_{6\times1}(\mathbb{R}_{max})$, où $B^t = \begin{pmatrix} \tau_1 & \tau_1 & \tau_1 & \epsilon & \epsilon & \epsilon \end{pmatrix}$;

– $U(k) = u(k)$;

– $F_1 \in \mathcal{M}_{6\times6}(\mathbb{R}_{max})$, où $F_1(k) = \begin{pmatrix} f_{x_4}(k) & \epsilon & & \cdots & & \epsilon \\ \epsilon & f_{x_5}(k) & \epsilon & & & \\ & \ddots & f_{x_6}(k) & \epsilon & & \vdots \\ \vdots & & & & \epsilon & \\ & & & \ddots & & \epsilon \\ \epsilon & & \cdots & & \epsilon & \epsilon \end{pmatrix}$;

– $F_2 \in \mathcal{M}_{6\times6}(\mathbb{R}_{max})$, où $F_2(k) = \begin{pmatrix} f_{x_1}(k) & \epsilon & & \cdots & & \epsilon \\ \epsilon & f_{x_2}(k) & \epsilon & & & \\ & \ddots & f_{x_3}(k) & \epsilon & & \vdots \\ \vdots & & & & \epsilon & \\ & & & \ddots & & \epsilon \\ \epsilon & & \cdots & & \epsilon & \epsilon \end{pmatrix}$;

– $k_i = \bigotimes_{j=1,j\neq i}^{3} m_{Q_j}$ pour tout i, $4 \leq i \leq 6$, sachant que $m_{Q_j} = \bigotimes_{l=1}^{k} 1 \otimes f_{x_j}(l)$;

– $Y(k) = y(k)$;

– $C_{k_i} \in \mathbb{R}_{max}^{1\times6}$ avec $C_{k_4} = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \tau_5 & \epsilon & \epsilon \end{pmatrix}$, $C_{k_5} = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \tau_5 & \epsilon \end{pmatrix}$, $C_{k_6} = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \tau_5 \end{pmatrix}$

4 Conclusion

Dans ce chapitre, nous avons introduit les patterns et workflow dans la première partie. Ensuite, nous avons cité brièvement quelques recherches relatives à la composition de services en se basant sur les patterns. Nous avons également définit les critères de la qualité de service de manière générale, puis nous avons précisé les quatre critères de la QoS utilisés dans notre travail. Nous avons montré également la façon dont le score associé à un processus composé d'un ou plusieurs patterns est calculé. La deuxième partie a été consacrée aux deux formalismes de modélisation utilisés pour modéliser les patterns, à savoir les RdPs et l'algèbre (max,+). Nous avons introduit chacun de ces deux outils en rappelant les différentes définitions et caractéristiques qui seront utiles pour notre travail. Dans la troisième partie, Nous avons présenté les six patterns essentiellement utilisés pour la composition de services. Ces patterns ont été représentés en modèles RdP P-temporisés, ensuite en équations mathématiques dans l'algèbre

(max,+). Nous avons terminé cette partie par une illustration d'un exemple de composition de patterns, où nous avons explicité ses équations (max,+).

Comme nous le verrons plus loin dans le chapitre 5, ces équations nous permettrons d'évaluer les performances et de vérifier et valider les processus de composition de services en analysant leurs propriétés qualitatives. Nous allons voir dans le chapitre 5 un scénario d'application de ces patterns et leurs équations associées au sujet de la composition de plusieurs services métiers distribués pour la sécurité routière dans les transports.

Dans le chapitre suivant, nous allons concrétiser nos travaux de recherche par l'élaboration d'une plateforme de collaboration de LBS. Cette plateforme, appelée TransportML, est le fruit de nos travaux de recherche effectués dans le cadre des projets Européens ASSET et TeleFOT. Lors de la collaboration et la composition de services, les patterns définis dans ce chapitre seront utilisés à grande échelle. Les critères de la QoS sont également pris en considération lors de la composition de services par la plateforme. Les performances de la composition de services évaluées à partir de cette plateforme seront comparées à celles évaluées, d'une part, à partir de l'étude analytique (modèles RdPs et équations (max,+)), et d'autre part, à celles obtenues à partir des expérimentations réelles menées sur le terrain. Le chapitre suivant est consacré donc à l'exposition et à la définition de l'architecture et les composants de la plateforme TransportML.

Chapitre IV

TransportML pour la collaboration des LBS

La plateforme TransportML est le résultat des travaux de recherches approfondies que nous avons menés dans le cadre du projet Européen ASSET. Il peut être considéré comme une interface standard entre les services de localisation exposant leurs données tout en respectant les principes définis par l'architecture SOA (*Service Oriented Architecture*). Plus précisément, TransportML permet l'interaction des services Web d'une façon automatique, ce qui signifie qu'il n'est pas nécessaire de définir un scénario durant la phase de développement. Dans ce chapitre, nous allons décrire en détail cette plateforme de collaboration des LBS. Nous allons également définir ses composants, son architecture, ainsi que le langage utilisé pour l'échange d'informations entre ses composants.

1 Description générale

Ces dernières années, grâce à la disponibilité omniprésente des technologies de communication sans fil et des appareils sans fil, des efforts considérables dans la recherche ont été effectués dans le domaine de la communication inter-véhicule. L'objectif de ces recherches est d'accroître la sécurité et le confort des conducteurs en relayant les informations d'un véhicule à l'autre. Á titre d'exemple, les futures véhicules équipés pourront anticiper et éviter les zones où la circulation est perturbée, se déplacer vers leur destination en empruntant les plus courts chemins tout en contournant des routes avec obstacles (travaux, neige, embouteillage, etc.), identifier le parking le plus proche avec des places disponibles [DBG+10]. Par ailleurs, l'émergence des infrastructures GNSS a encouragé des recherches approfondies dans le développement des services et des applications à base de géolocalisation. Pour développer de tels services, différents éléments sont indispensables, à savoir, les appareils mobiles, les réseaux de communication, les techniques de positionnement et les fournisseurs de services et d'informations. Les périphériques tels que les ordinateurs portables, les PDA, les smartphones et les téléphones portables, sont

utilisés pour interroger et échanger avec ces services et traiter les résultats. Le réseau mobile est un autre composant impératif pour transférer les demandes des utilisateurs aux fournisseurs de services et d'envoyer les informations demandées aux utilisateurs finaux. Les composants de positionnement sont utilisés pour déterminer l'emplacement de l'utilisateur, afin de personnaliser les informations requises et de fournir le résultat le plus pertinent par rapport à l'emplacement de l'utilisateur. Les fournisseurs de services sont implémentés dans une infrastructure prédéfinie (c.-à-d. Internet) et offrent un certain nombre de services différents aux utilisateurs. Une plateforme est donc nécessaire pour faciliter les échanges et les interactions entre tous ces dispositifs et offrir aux utilisateurs l'accès aux services pertinents.

Plusieurs plateformes ont été développées pour des applications mobiles d'une façon générale comme indiqué dans [BPG03, ACBCB+11] ainsi que pour les LBSs [KL03, HR04]. Cependant, le développement des LBS interopérables pour le transport routier est délaissé et aucune de ces plateforme ou peu, ne se soucie de cette problématique si importante. En plus de la question d'interopérabilité traitée par ces plateformes, les LBS introduisent de nouvelles exigences pour le transport routier, telles que la nécessité d'interaction et de collaboration entre les services et la quantité d'informations existantes qui doivent être traitées efficacement par ces plateformes. Dans ce chapitre, nous développons une plateforme de collaboration, d'interaction et d'interopérabilité des LBS en tenant compte de différentes exigences de ces services.

1.1 Besoins et exigences

Actuellement, chaque zone géographique peut bénéficier d'un ensemble de services locaux. Ces services sont généralement fournis par les collectivités locales, tels que le service de déneigement, les travaux routiers, la collecte des déchets, le service d'urgence, etc. Chacun de ces services remplie souvent sa tâche indépendamment du reste des services. Or, les informations recueillies par un service peuvent être très utiles pour les autres services. Palier à ce manque de communication peut se révéler crucial dans une situation d'urgence. En effet, partager ces informations permettra aux utilisateurs de la route d'être mieux informés sur le contexte de conduite, ce qui les aidera à prendre les décisions appropriées et, par conséquent, accroître et améliorer la sécurité routière. Les technologies de communication sans fil, associées au GNSS et aux systèmes d'information, sont parmi les moyens utilisés pour développer l'interopérabilité dans les systèmes de transport. Par ailleurs, l'évolution des infrastructures orientées services permet le partage d'informations entre les applications sur Internet. C'est dans cette optique que TransportML a été développé. En effet, TransportML constitue une plateforme collaborative et distribuée, permettant la coopération et l'interaction des services.

La plateforme TransportML est accessible directement par les utilisateurs reliés à Internet *via* une technologie de communication, par exemple GPRS qui a été identifié pour être approprié pour le déploiement des applications à base de positionnement [Vir01]. GPRS est une technique de communication cellulaire dont l'infrastructure est disponible et bien répartie dans la plupart des pays européens. Cette technologie fournit une gamme de communications très élevée sans coût de déploiement. Par ailleurs, la communication entre les services Web est généralement réalisée en utilisant Ethernet. La force de ce moyen de communication est qu'il est standard et ne dépend d'aucune plateforme ou langage. De ce fait et en raison de la nature même du Web, l'interopérabilité est plus simple à mettre en place. Ceci est encore plus intéressant parce que TransportML vise à standardiser l'interaction entre des services hétérogènes.

1.2 Architecture

L'architecture d'un service Web est composée de trois entités principales : le fournisseur de service, le demandeur de service et l'annuaire de service comme décrit dans le chapitre 1, §1.2. Le fournisseur de service héberge un service Web qui peut être invoqué par le demandeur de service (c.-à-d. client ou utilisateur) à travers des requêtes respectant le format SOAP basé sur XML. Pour rendre un service Web accessible et faciliter sa découverte, le fournisseur de service doit l'enregistrer auprès d'un annuaire de service (p. ex. UDDI). Ce dernier peut être interrogé par les clients, *via* TransportML, afin de leur permettre d'identifier les services qui peuvent satisfaire leurs demandes. Cependant, cette architecture ne peut pas être directement utilisée, pour développer les services à base de géolocalisation, dans le domaine de transport routier. La plateforme TransportML est donc une extension de l'architecture SOA facilitant la collaboration entre services routiers.

La figure IV.1 présente l'architecture générale de TransportML. Cette architecture est composée d'une boîte à outils (que nous appelons Middleware TransportML dans la figure) de coordination, de coopération et de synchronisation entre les services ; d'un annuaire TML_UDDI permettant de stocker les informations sur les services ; d'un journal de qualité de service - QoSLog - qui permet d'enregistrer les informations sur la qualité des services invoqués. Les différents acteurs interagissant avec cette plateforme sont les fournisseurs de services et les clients finaux (ou utilisateurs). La mise en oeuvre de la plateforme gère les différentes interactions entre les acteurs. Ces interactions sont assurées par une couche de communication basée sur les réseaux de communication et les systèmes de positionnement. Dans l'architecture TransportML, les services sont décrits d'une façon sémantique et publiés dans l'annuaire TML_UDDI.

Les fournisseurs de services enregistrent des services conformes à la plateforme Trans-

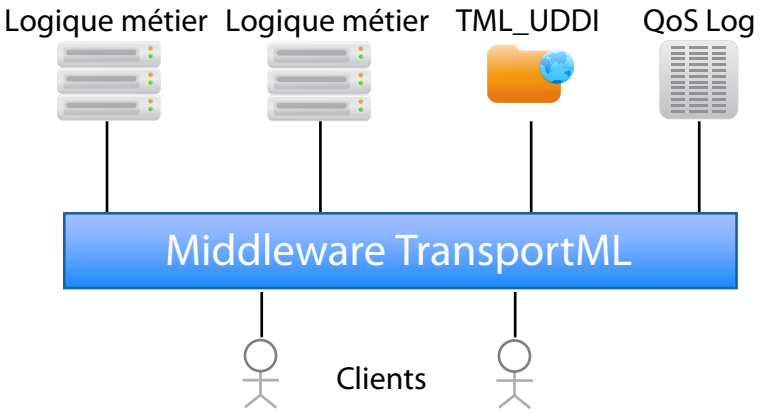


Figure IV.1 – Architecture de TransportML

portML. Cette publication ne contient pas l'ensemble des informations relatives aux services, mais une description permettant aux demandeurs de déterminer les services appropriés pouvant répondre d'une façon efficace à leurs requêtes. Dans la section suivante nous allons définir le format d'échange entre les différents composants de la plateforme TransportML.

2 Protocole de communication TML

2.1 Description

Dans la plateforme TransportML, nous avons choisi de définir et d'utiliser un nouveau langage appelé Tranport Markup Language (TML) qui est inspiré des languages mentionnées précédemment dans le chapitre 1, §1.3.1. Le choix de ce nouveau langage est dû au fait que des langages comme DATEX II, tpegML ou encore GML sont des solutions lourdes fournissant de multiples fonctionnalités qui ne sont pas utiles pour les besoins particuliers de la plateforme TransportML. D'autre part, ces langages n'ont pas les caractéristiques spécifiques requises par la plateforme afin de soutenir la composition automatique de services. Néanmoins, cette plateforme se base sur quelques fonctionnalités qui peuvent être fournies par ces langages. Dans la plateforme TransportML, le scénario de composition de service est élaboré à la volée en s'appuyant sur la requête basée sur le format TML. Par conséquent, TML est un format utilisé, non seulement, pour l'échange d'information géographique, mais aussi pour les requêtes de l'utilisateur. Toutefois, DATEX II et tpegML sont simplement des formats d'échange d'informations

géographiques et ils ne sont pas adaptés pour ce type de requête. Par conséquent, ils ne sont pas utilisables pour les échanges entre les différents acteurs de la plateforme TransportML.

Les différentes fonctions permettant de gérer et de définir les composants de ce format sont fournis par notre bibliothèque TransportML. Le diagramme de classe est donné dans la figure IV.2.

2.2 Le langage TML

TML est un langage de transport définissant un format d'échange entre les entités de la plateforme TransportML. Un langage standard de communication unifié est une condition préalable pour l'interaction automatique des services. En effet, le scénario est élaboré à la volée en examinant un document TML correspondant à la requête de l'utilisateur. Comme on peut le voir dans la figure IV.3, un document TML est un document XML ayant un élément racine <tml_document>. Cette racine a un élément enfant <vehicle> qui correspond aux caractéristiques du véhicule. Un autre élément <criteria> qui renseigne des préférences de l'utilisateur. Nous signalons qu'il est possible d'ajouter, dans cette section, d'autres critères et préférences. La dernière section correspond au type de la requête <route>, qui dans notre cas, spécifie la demande d'un itinéraire par un utilisateur. Nous nous référons également à cet élément comme une section du document TML. Ces différentes sections sont détaillées dans ce qui suit.

2.2.1 La section <vehicle>

La section <vehicle> se compose des éléments suivants :

- <type> qui défini le type du véhicule et prend trois valeurs possibles : voiture, poids lourd ou moto,
- <weight> correspond au poids du véhicule,
- <TDG> C'est l'acronyme de *Transportation of Dangerous Goods*, c'est une valeur booléenne égale à *true* si le véhicule transporte de la matière dangereuse,
- <dimensions> est les dimensions du véhicule avec deux éléments :
  - <width> est la largeur du véhicule,
  - <height> est la hauteur du véhicule.

2.2.2 La section <criteria>

Dans la section <criteria> nous définissons les préférences des utilisateurs pour les critères de la QoS. Cette section contient les éléments correspondants à ces critères qui sont supportés

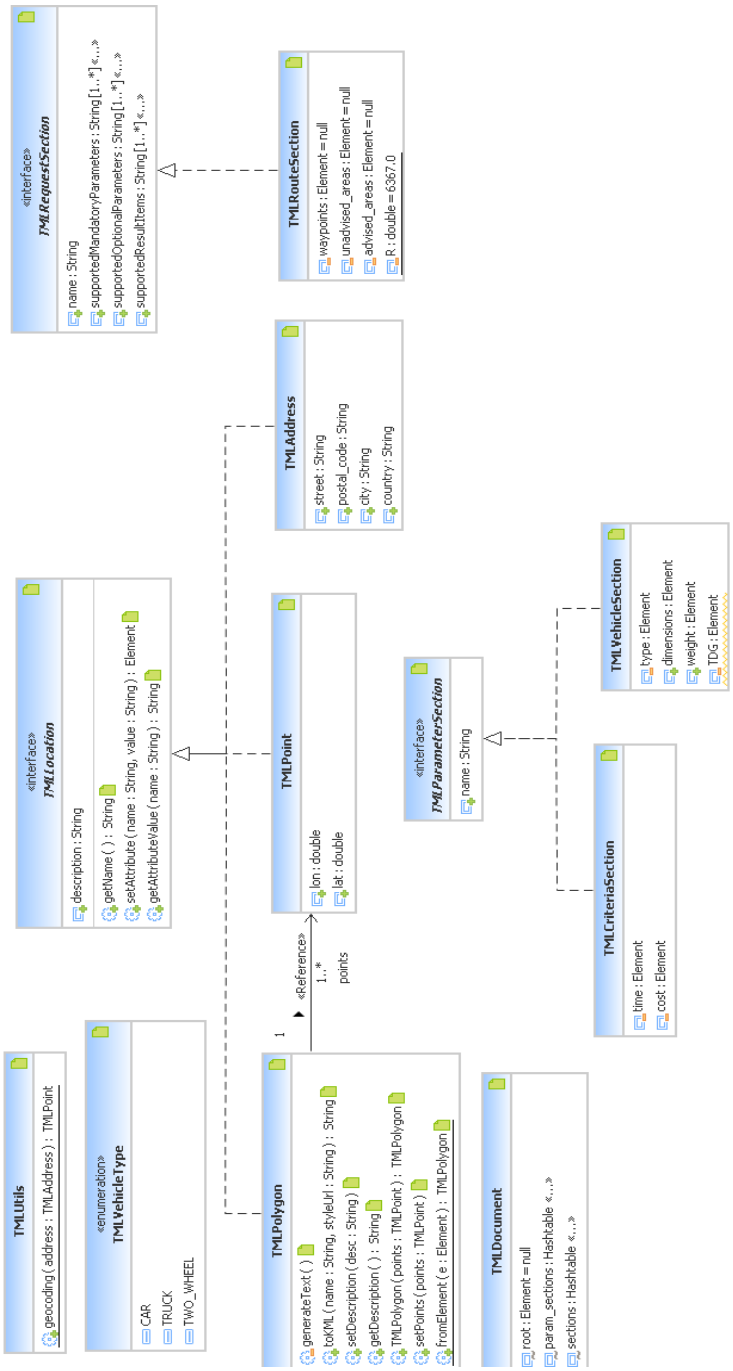


Figure IV.2 – Le diagramme de classe TML

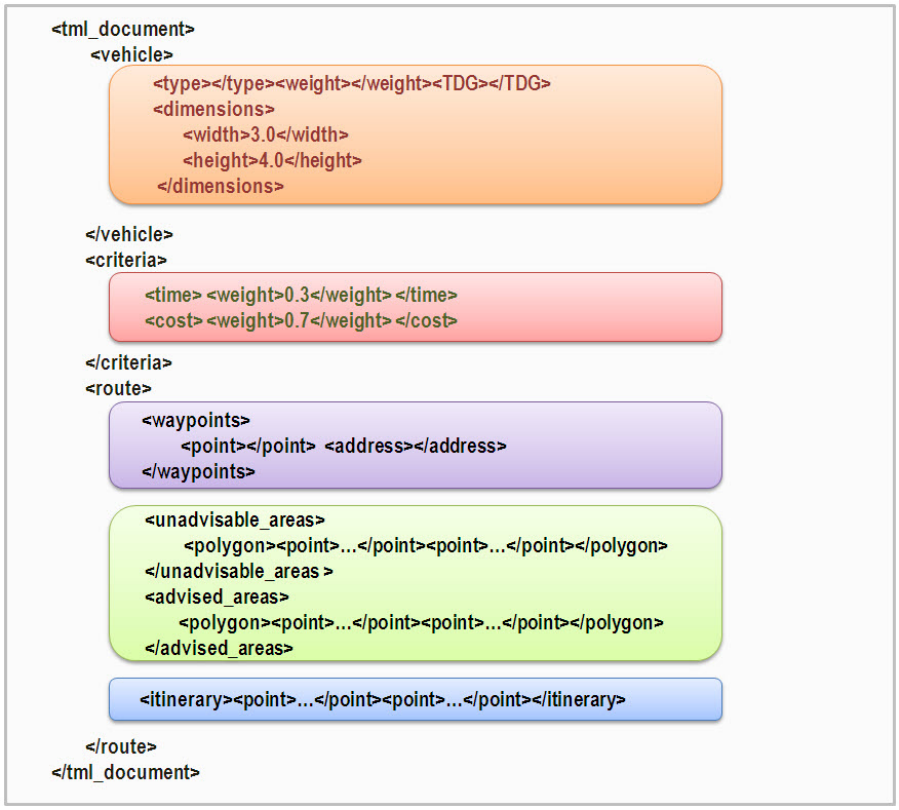


Figure IV.3 – Structure d'un document TML

par la plateforme. Toutefois, comme vu précédemment le système n'est pas limité à ces critères, nous pouvons éventuellement en ajouter d'autres dans cette section.

2.2.3 La section <route>

La section <route> contient plusieurs éléments correspondant, soit à des paramètres obligatoires ou optionnels, soit aux résultats de la requête. Dans l'exemple choisi, l'élément <waypoints>, correspond aux points de départ et de destination. Les éléments <advised_areas> et <discouraged_areas> correspondent à des paramètres optionnels qui peuvent être remplis par des services tiers, avant le calcul d'itinéraire, afin d'apporter une valeur ajoutée. Enfin, le dernier élément, <itinerary>, contient la réponse de la requête de l'utilisateur.

Comme illustré dans la figure IV.3, la plateforme TransportML supporte trois façons pour représenter une zone géographique :

- <point> : Un point est caractérisé par ses coordonnées GPS (exprimés dans la norme WGS84) : latitude, longitude et altitude. C'est le moyen le plus précis et le plus efficace pour exprimer un point ponctuel. Un point est représenté dans le document TML comme suit :
  - <point> latitude, longitude, altitude </point>
  - Exemple : <point> 47.4356,6.4765,0 </point>
- <adresse> : Une adresse est caractérisée par les attributs suivants : rue, code postal, ville, pays. La plateforme TransportML dans sa version actuelle est en mesure de traiter seulement les coordonnées GPS. Pour faire face à cette limite, nous avons défini des fonctions dans la librairie TransportML, permettant la conversion d'une adresse postale en coordonnées GPS. Cette conversion est connue sous le nom de *geocoding*. L'opération inverse permet de convertir les coordonnées GPS en adresse postale, cette opération est appelée *reverse geocoding*. Une adresse est représentée comme suit :
  - <adresse> rue, code postal, ville, pays </address>
  - par exemple : <adresse> 3 rue Jean Jaurès, 90000, Belfort, France </address>
- <polygon> : Un polygone permet de représenter une zone géographique. Il est caractérisé par au moins trois points précis (c.-à-d. <point> ou <adresse>). La liaison de ces points forme un polygone qui est représenté comme suit :
  - <polygon> <point> ...</point> <adresse> ...</adresse> </polygon>
  - par exemple : <polygon> <point> 6.4765,47.4356,0 </point> <point> 6.4865,47.4356,0 </point> <point> 6.4765,47.5356,0 </point> </polygone>

Les requêtes des utilisateurs sont fournis au format TML comme illustré dans l'exemple de

```
- <tml_document>
  - <vehicle>
      <type>CA</type>
      <weight>5.3</weight>
      <TDG>0</TDG>
    - <dimensions>
        <width>3.0</width>
        <height>4.0</height>
      </dimensions>
    </vehicle>
  - <criteria>
    - <time>
        <weight>0.3</weight>
      </time>
    - <cost>
        <weight>0.7</weight>
      </cost>
    </criteria>
  - <route>
    - <waypoints>
        <address>6 bd anatole france,90000, belfort,france</address>
        <address>Faubourg de Brisach,90000, belfort,france</address>
      </waypoints>
    - <unadvised_areas>
      - <polygon ratio="9999999.0">
          <point>47.633412251132214,6.849121570849093,0</point>
        </polygon>
      </unadvised_areas>
    - <advised_areas>
      - <polygon ratio="0.5">
          <point>47.64052594749587,6.85993409169896,0</point>
        </polygon>
      </advised_areas>
    - <itinerary>
        <point>47.6412353,6.8465221,0</point>
      </itinerary>
    </route>
  </tml_document>
```

Figure IV.4 – Exemple d'un document TML

la figure IV.4. Comme on peut le voir dans cet exemple, l'utilisateur a fourni les paramètres obligatoires (c'est à dire l'emplacement de départ et de destination). En outre, l'utilisateur spécifie les caractéristiques de son véhicule, ainsi que ses préférences en termes de critères de qualité de service. Nous remarquons également que les éléments correspondant à des paramètres optionnels (<discouraged_areas> et <advised_areas>) sont spécifiés mais vides. Cela signifie que l'utilisateur est intéressé par ces paramètres facultatifs et que la plateforme TransportML devrait essayer de trouver des services tiers adéquats et qui seront en mesure de fournir des informations liées à ces sections. En parallèle, ce même raisonnement peut s'appliquer à l'élément <itinerary> qui correspond à la demande de l'utilisateur.

3 Description du moteur de TransportML

Le moteur de la plateforme TransportML agit comme un moyen de communication qui peut être utilisé par des utilisateurs mobiles. Il contient les cinq composants principaux suivants comme indiqué dans la figure IV.5 : le service d'enregistrement, le gestionnaire de requête, le service d'orchestration et le service sélecteur. Les utilisateurs interagissent avec le gestionnaire de requête. Ils sont généralement les usagers qui souhaitent récupérer des informations liées à leur contexte. Pour atteindre cet objectif, les utilisateurs envoient leurs requêtes qui seront par la suite transmises au gestionnaire de requêtes. Toutes les requêtes envoyées à la plateforme seront dans un premier temps traitées par ce gestionnaire de requête. Les fournisseurs de services enregistrent, *via* le service d'enregistrement, les informations relatives à leurs services dans l'annuaire TML_UDDI. Chaque service enregistré pourrait être invoqué en se basant sur ces informations. Une fois enregistrés, les services peuvent être sélectionnés pour satisfaire les demandes des utilisateurs. Pour ce faire, on fait appel dans un premier temps au générateur de pattern. Ce dernier va générer le plan du scénario qui est une agrégation de tâches où chaque tâche est traitée par un service. Cette agrégation de tâches est envoyée au sélecteur qui va sélectionner les services adéquats en se basant sur des critères de la QoS . Il fournira ensuite une liste de services sélectionnés répondants à la requête. Cette liste de services est alors envoyée au service d'orchestration qui se chargera d'exécuter le scénario. Le service d'orchestration va donc consulter les informations, fournies à l'inscription et stockées dans TML_UDDI, pour invoquer les services sélectionnés.

Les interactions entre les composants du moteur TransportML gérant les requêtes des utilisateurs sont réalisées à travers les étapes suivantes : TransportML reçoit une requête envoyée par un utilisateur *via* son équipement. Cette requête est formatée par le gestionnaire de requête

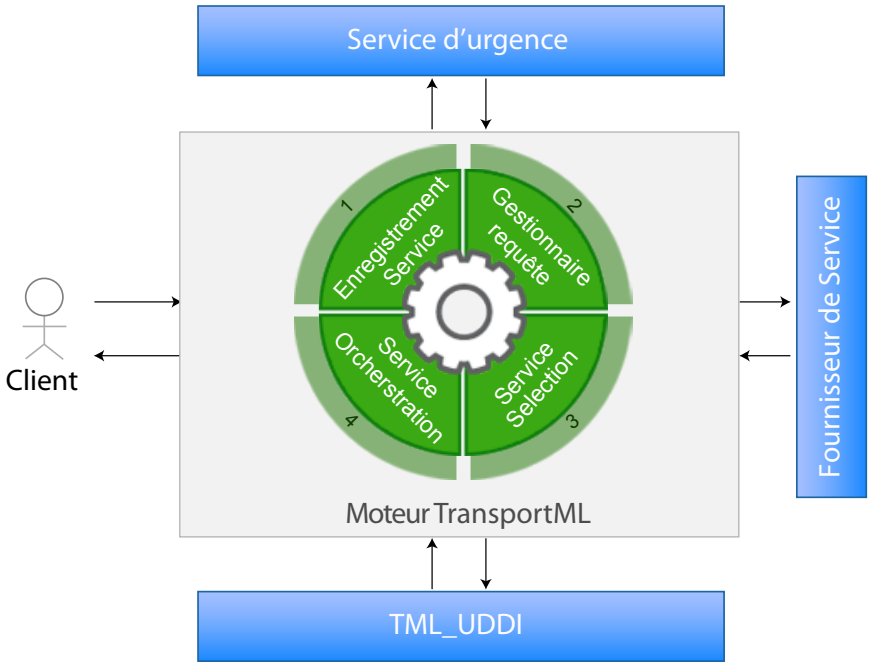


Figure IV.5 – Les composants de TransportML

au format d'échange utilisé par la plateforme. La requête est ensuite transmise au service d'orchestration. Ce dernier va invoquer les services sélectionnés et fournis au préalable par les deux composants : le générateur de pattern et le sélecteur, en leur fournissant les documents TML (décrit dans la section 2) en entrée. Dès que le moteur d'orchestration a reçu les documents résultants des services invoqués, il les fusionne en vue d'obtenir un document TML unique. Le service d'orchestration construit dans cette étape un document contenant à la fois les paramètres fournis par l'utilisateur, et les paramètres complétés par les services tiers. Dans les sections suivantes, nous allons décrire chaque composant de la plateforme.

3.1 Enregistrement de services

Un service peut être enregistré dans la plateforme *via* le service d'enregistrement. Ce dernier est un service Web qui fournit deux méthodes : *boolean registerService(String tmlsd)* et *boolean unregisterService(String WSDL_URL)*. Ces méthodes renvoient une valeur booléenne égale à *true* si l'opération a été exécutée avec succès, et *false* sinon.

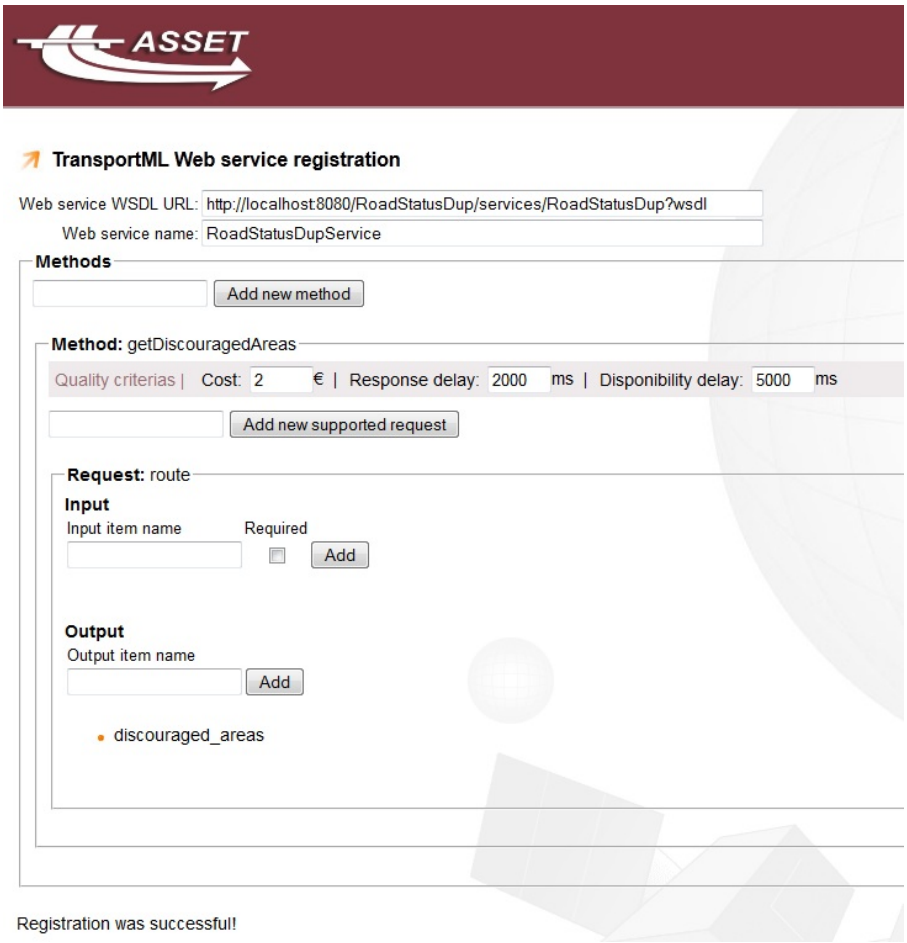


Figure IV.6 – Interface Web pour l'enregistrement d'un service

3.1.1 Description de service

Un service est enregistré dans la plateforme TransportML par le fournisseur de ce service à travers un formulaire Web illustré par la figure IV.6. Lorsque le formulaire est rempli, les informations sont automatiquement enregistrées dans l'annuaire TML_UDDI. Cette application exécute en arrière plan un client de service Web qui est appelé lors de la soumission du formulaire afin de contacter le service d'enregistrement.

Lors de l'enregistrement du service auprès de la plateforme, certaines informations concernant les fonctionnalités de service doivent être fournies. Ces informations ne peuvent pas être incluses dans le fichier WSDL du service. Pour remédier à cette incompatibilité, nous avons introduit une description complémentaire appelée la *Description de Service TML* (TML SD, *TML Service Description*), que nous décrivons dans la prochaine section.

```
<TMLServiceDescription>
    <WSDL_URL>asset-set.utbm.fr:8080/ItineraryService.wsdl</WSDL_URL>
    <ServiceName>ItineraryService</ServiceName>
    <Methods>
        <Method name= "getItinerary">
            <SupportedRequests>
                <Request name= "route">
                    <Input>
                        <Item item_name= "unadvised_areas" required= "false" />
                        <Item item_name= "advised_areas" required= "false" />
                        <Item item_name= "waypoints" required= "true" />
                    </input>
                    <output>
                        <Item name= "itinerary" />
                    </output>
                </Request>
            </SupportedRequests>
            <Criteria>
                <cost />
                <time />
                <reliability />
                <disponibility /> ...
            </Criteria>
        </Method>
    </Methods>
</TMLServiceDescription>
```

Figure IV.7 – Structure de la déscription de service

3.1.2 La description de service TMLSD

Une description de service TransportML est un document XML avec une syntaxe simple, comme on peut le voir dans la figure IV.7. La bibliothèque TransportML comprend une couche d'abstraction permettant la création et la gestion du TML SD. La description de service TML est une description complémentaire au WSDL où l'on ajoute des informations sur des fonctionnalités spécifiques aux services liés à la plateforme TransportML. La figure IV.7 présente le TML SD correspondant à la description du service *Itinerary*.

Une description du service TML comprend les informations suivantes :

- l'emplacement du service ou l'URL de sa description WSDL,
- le nom du service,
- la méthode du service respectant le format de la plateforme. Pour chaque méthode, nous spécifions :
  - le type de la requête (p. ex. route),
  - les éléments pris en charge en entrée (p. ex. *waypoints*, *advised_areas*, *discouraged_areas*) et s'ils sont obligatoire ou non,
  - le résultat en sortie (p. ex. itinéraire).

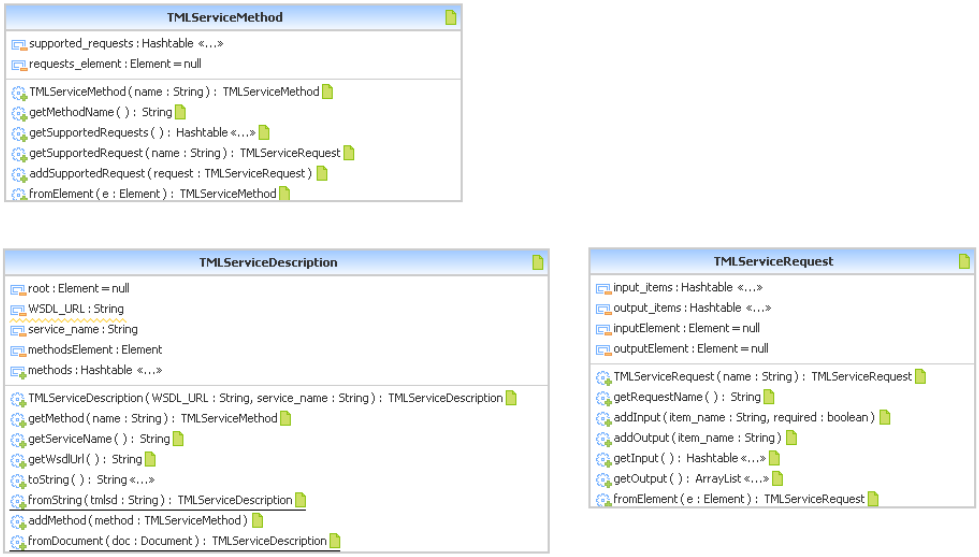


Figure IV.8 – Le diagramme de classe pour la description de service TML SD

– les valeurs de la qualité de services (p. ex. coût, temps). Nous rappelons que nous avons défini dans le chapitre 3, §1.3 quatre critères de qualité de service pris en compte dans notre étude. Ces derniers correspondent aux coût (c.-à-d. l'élément <cost>), la durée (c.-à-d. l'élément <time>), la fiabilité (c.-à-d. l'élément <reliability>) et la disponibilité (c.-à-d. l'élément <availability>). Dans la description de service, nous précisons que tous les critères supportés par la plateforme sont définis dans ce fichier, mais seuls le coût et la disponibilité sont renseignés par le fournisseur du service. Les autres critères sont calculés à chaque appel du service, et les valeurs sont enregistrées dans un fichier historique contenant toutes les invocations précédentes avec les valeurs correspondantes à chaque critère. Il est à noter que notre système est extensible et donc par conséquent, il est possible d'ajouter d'autres critères concernant la qualité de services.

Dans la figure IV.8, nous représentons les différentes classes permettant la description d'un service. Ces classes permettent de définir les différentes sections dans le document TML SD. Ainsi, elles définissent les différentes méthodes et fonctions pour générer les différents éléments de cette description.

En effet, le scénario est déterminé au moment de l'exécution, en utilisant la description de service détaillée dans la figure IV.7. En conséquence, TransportML est dynamique et extensible, du fait qu'il permet à de nouveaux services de rejoindre la communauté et de bénéficier de son bus de communication d'une manière simple. Nous allons définir dans la section suivante le composant permettant la gestion des requêtes des utilisateurs.

3.2 Le gestionnaire de requêtes

Le gestionnaire de requête *Request handler* est le point d'entrée de la plateforme TransportML. Sa capacité de traiter toute requête respectant le format TML l'a rendu universel. Ce gestionnaire de requête est implémenté comme un service Web à l'aide de l'API JAX-WS et fournit une méthode *String handleRequest(String tml)*. Cette méthode est synchrone et renvoie le document TML résultant sous forme d'une chaîne de caractères. Fondamentalement, le document renvoyé par cette fonction est le même que la requête de l'utilisateur, où les éléments des sections, *vehicle*, *criteria*, et *waypoints* sont renseignés.

3.3 Service de sélection

Une des tâches du moteur d'interaction de service TransportML est d'élaborer, au moment de l'exécution, des scénarios de composition satisfaisant les demandes des utilisateurs. Pour réaliser cette tâche, le moteur de l'interaction doit prendre en considération les dépendances inter-services. Par exemple, il est possible que le service métier nécessite le résultat d'un autre service pour réaliser sa tâche. En conséquence, nous devrions fournir des moyens exprimant les dépendances de permettre aux fournisseurs de services de renseigner cette information lors de l'enregistrement de leurs services dans l'annuaire de la plateforme. Il est également important que cette dépendance soit facilement compréhensible par la plateforme TransportML.

Le prototype développé gère ces dépendances inter-services. En effet, lors de l'enregistrement d'un service, ces dépendances sont exprimées d'une façon implicite, en spécifiant les paramètres en entrée du service et les paramètres en sortie. Cette dépendance est considérée entre les services sélectionnés pour compléter les paramètres des sections et sous-sections choisies afin de compléter la section correspondant au résultat. Pour ce faire, nous incluons des informations concernant les prérequis de services dans leur description sémantique. Le service de sélection se base sur cette description pour établir l'ordre d'appel entre les services et aussi pour déterminer les services concurrents et qui sont susceptible de répondre à une tâche.

Le processus établit par ce composant est décrit dans ce qui suit. En effet, le service de sélection s'occupe de la sélection des services pour chaque tâche dans un scénario décrit dans un document TML. Ce document est fournit par le gestionnaire de requêtes. Le service de sélection parcourt donc dans un premier temps le document TML, et pour chaque section rencontrée, il vérifie sur l'annuaire TML_UDDI les différents services susceptibles de traiter la requête. La deuxième phase consiste à récupérer l'historique de qualité de services issu des invocations précédentes. Il lit également la description du service afin de récupérer les paramètres non fonctionnels (p. ex. coût) des services. Les valeurs de la QoS de tous les services candidats sont

connues à partir de cette étape. La tâche suivante repose sur le calcul du score de chaque service en utilisant les formules décrites dans le chapitre 3, §1.3. Pour finir, ce service de sélection va établir une liste des services finaux sélectionnés pour exécuter chaque tâche dans le scénario. Cette liste sera communiquée au composant suivant qui est le service d'orchestration.

3.4 Service d'orchestration

Le service d'orchestration décrit d'une façon automatique la coordination et la gestion des services. Habituellement, le service d'orchestration est réalisé par l'élaboration d'un processus métier avec l'exécution des langages tels que WS-BPEL. En effet, en utilisant BPEL les services, à orchestrer, sont désignés au moment du développement, ce qui n'est pas l'objectif de la plateforme TransportML (les services sont sélectionnés au moment de l'exécution). Surtout que, BPEL ne peut pas satisfaire une orchestration dynamique et extensible telle qu'elle est requise par le moteur TransportML. Par conséquent, afin d'orchestrer les services d'une façon dynamique, un service d'orchestration est développé. Ce service décrit la coordination et la gestion des services sélectionnés.

Le service d'orchestration développé prend en entrée une requête utilisateur formatée en un document TML et détermine quels services devraient être contactés afin de satisfaire la demande au moment de l'exécution. En outre, les services impliqués dans TransportML ne sont pas connus au moment du développement, car ils sont enregistrés auprès du service enregistrement de la plateforme. Le service d'orchestration se base sur la liste des services fournis par le service de sélection et établit un scénario d'exécution. Une fois que le scénario d'exécution est déterminé, le service d'orchestration invoque les services et fusionne les résultats fournis par ces services.

Le service orchestration de la plateforme TransportML est implémenté comme un client de service Web qui invoque les services Web enregistrés dans la plateforme. Il se base, pour cela, sur les informations telles que l'URL du WSDL du service, le nom du service et le nom de la méthode. Toutes ces informations sont fournies par le fournisseur de service au moment de l'enregistrement. Il est à noter que les services Web sont invoqués de façon asynchrone afin d'optimiser le temps de réponse. Ce composant prend soin de fusionner les documents émis par les divers services et envoie ce document résultant au service gestionnaire de la requête qui s'occupe d'effectuer l'opération inverse en envoyant le résultat final sous forme d'un document TML à l'utilisateur.

4 Discussion sur la plateforme TransportML

Comme expliqué précédemment, ce prototype est une preuve de concept de la faisabilité d'une plateforme d'interopérabilité des LBS dans le domaine du transport routier. Un des atouts et points forts de cette plateforme repose principalement sur sa possibilité de faire collaborer des services distribués et hétérogènes d'une manière automatique. En effet, plusieurs types de services possèdent des informations hétérogènes qui peuvent être enregistrées et stockées selon plusieurs formats différents. Cette hétérogénéité rend la communication inter-services difficile voire impossible. Pour faire face à ce problème, le langage standard de communication TML (c.-à-d. Transport Markup Language) est proposé. De même, un moteur d'interaction automatique de services est déployé afin d'élaborer à la volée des scénarios de collaboration fondés sur des requêtes au format TML. En effet, le langage TML fournit un protocole qui doit être respecté par tous les acteurs (p. ex. clients, services métiers). L'avantage de ce langage réside dans le fait qu'il peut contenir toute information liée à la route et délivrée par diverses entités locales. Les scénarios d'interaction de services peuvent être également conclus à partir des requêtes au format TML ce qui permet une composition automatique des services.

Le fait qu'il n'y ait pas de scénario prédéfini rend la plateforme très extensible, et de nouveaux services peuvent facilement rejoindre la plateforme sans aucune modification. Pour joindre la plateforme, les entités locales (p. ex. collectivités locaux, fournisseur de service) devront simplement exposer leurs informations liées à la route en utilisant les services Web. Ces derniers sont désormais le moyen standard d'exposition de données sur Internet. Les fournisseurs de services doivent ensuite enregistrer leurs services dans l'annuaire TML_UDDI, afin qu'ils puissent être invoqués par la plateforme. Ces services devront également avoir en entrée le document TML et produire en sortie des documents TML valides. Ces documents, qui sont des simples documents XML avec un format bien défini, peuvent être facilement analysés et générés à l'aide des librairies standard XML. Toutefois, afin de faciliter cette tâche, TransportML fournit une librairie permettant de générer ces documents. Cette librairie fournit également une variété de fonctionnalités supplémentaires, à savoir le *geocoding*, *reverse geocoding*, et d'autres fonctions permettant de gérer les descriptions des services. Elle s'occupe également de la conversion d'un document TML au format KML (c.-à-d. le langage géographique utilisé par Google). Ce qui permet d'afficher et de visualiser le résultat (p. ex. itinéraire, polygones, points) à l'aide des outils gratuits fournis par Google, comme Google Map et Google Earth.

Une exigence importante pour la composition automatique des services Web est la description des services et leurs capacités. En effet, dans la composition automatique, ni les services impliqués ni le scénario de composition ne sont identifiés lors de l'étape de développement. En

conséquence, le scénario doit être établi à la volée pour satisfaire la requête d'un utilisateur, ainsi les services impliqués dans ce scénario devront être découverts et choisis au moment de l'exécution. Le service de sélection est basé sur divers critères de QoS pour effectuer les choix les plus pertinents.

Les documents TML ont été adaptés afin de contenir des informations concernant la QoS exigée par les utilisateurs. Nous pouvons également ajouter d'autres critères par exemple la priorité à certaines demandes, afin de permettre à la plateforme d'offrir une meilleure qualité par le traitement des demandes les plus urgentes d'abord et aussi prendre en considération l'urgence tout en élaborant des scénarios d'interaction.

5 Conclusion

Dans ce chapitre, nous avons présenté une plateforme d'interaction et de collaboration automatique des LBS. La plateforme TransportML est extensible, permettant à de nouveaux services de s'enregistrer et bénéficier ainsi de son support de communication. La plateforme est dynamique, car aucun scénario n'est prédéfini au moment du développement.

Le développement de la plateforme adapté aux besoins particuliers des LBSs est essentiel pour la création ainsi que pour le déploiement rapide de ce type de services. La plateforme traditionnelle axée sur l'informatique distribuée ne fournit pas d'API appropriée et des services d'infrastructure mais plutôt, elle offre des outils pratiques qui cachent la complexité liée aux tâches de positionnement, par exemple, les aspects de la qualité et la confidentialité.

Afin de couvrir tous les aspects des LBS, ce document a étudié les caractéristiques des LBS et leurs exigences pour la plateforme TransportML. Cette plateforme a donc été développée en tenant compte de ces exigences. Son architecture, ses composants et son mode de fonctionnement ont été illustrés. Ce prototype a été utilisé pour un scénario d'urgence que nous détaillons dans le chapitre 5. Une description de ce scénario ainsi que les expérimentations et une étude de performance sont établies dans ce chapitre.

Chapitre V

Expérimentations et résultats

Dans ce chapitre, nous présentons les différents scénarios que nous avons développé concernant la sécurité routière dans le cadre du projet Européen ASSET [1] FP7 (Advanced Safety and Driver Support for Essential Road Transport). Plus précisément, nous allons décrire le cadre de notre travail de recherche dans la section 1. Dans la section 2, nous présentons l'architecture générale adoptée par tous les scénarios développés. Ces scénarios sont exposés en détail dans la section 3. L'approche proposée dans cette thèse est illustrée par une étude de cas dans la section 4. Finalement, des tests et des expérimentations réelles sur le terrain sont présentés et rapportés dans la section 5. Nous présentons également dans la même section, l'implémentation du scénario proposé dans l'étude de cas ainsi que des différents résultats de simulation. Nous terminons le chapitre par une synthèse des résultats obtenus.

1 Cadre d'application : description du projet

L'élargissement de l'Union Européenne (UE) ainsi que l'accroissement de la mondialisation et de la concurrence poussent à une meilleure utilisation des ressources dans le domaine des transports. En effet, il existe un potentiel considérable pour accroître la pérennité des transports en termes de sécurité, de réduction de nombre d'accidents, de fluidité, d'efficacité, de protection des infrastructures et d'économie d'énergie. L'UE s'est fixée donc comme objectif l'amélioration des conditions du trafic routier pour assurer une sécurité plus importante dans le domaine du transport. L'UE a lancé différents projets ayant comme thématique principale la sécurité routière. Le projet Européen ASSET est un de ces projets qui a commencé en juillet 2008 et s'est terminé en décembre 2011. Il est supporté par la commission Européenne et également labellisé par le pôle de compétitivité *véhicule du futur* [2] en matière de *véhicules et réseaux intelligents.*

1. www.project-asset.com/
2. http ://www.vehiculedufutur.com/

Ce projet se concentre sur le développement et l'implémentation d'une approche holistique afin d'améliorer la sécurité routière et accroître l'efficacité des transports routiers pour la protection des conducteurs, des véhicules ainsi que des infrastructures routières. Pour ce faire, il se focalise sur l'assistance au conducteur, l'accroissement et la facilité d'accès aux connaissances de son contexte et l'amélioration de son comportement. Cependant, des solutions en temps réel tenant compte de l'état du trafic, de la circulation et d'autres incidents sont mises en oeuvre par le biais des capteurs, des technologies de communication et de géopositionnement, des infrastructures, des systèmes intelligents et des interfaces homme machine appropriées.

ASSET est un consortium de 19 partenaires européens dont sept universités, trois instituts de recherche, huit entreprises et un organisme administratif représentant 12 pays européens. Le consortium comprend également un partenaire Africain (Tanzanie) et un partenaire asiatique (l'Inde). Les partenaires rapportent leurs différentes contributions dans le projet ASSET. Chaque partenaire est impliqué dans un ou plusieurs workpackages. A titre d'exemple, un partenaire Allemand a développé, entre autres, un système de pesée de véhicules en mouvement permettant de peser précisément le poids des camions lors de leurs passages sur une plateforme de capteurs installée dans la chaussée. Le partenaire finlandais (VTT) ont conçu et développé un système de caméra pour surveiller le comportement et l'état du conducteur. Le système développé permet également de détecter le port ou non de la ceinture de sécurité. En tant que partenaire français, notre rôle dans ce projet est de concevoir et développer une plateforme d'interopérabilité, de coopération et de collaboration de différents services métiers distribués à des fins de sécurité dans le transport routier. Cette plateforme, appelée TransportML, sert également d'outil d'échange d'information et de communication des services distribués à base de localisation (LBS) en utilisant les technologies de communication sans fil associées aux systèmes de navigation par satellite (GNSS) et aux standards des services Web. La plate-forme dotée de ces technologies permet aux utilisateurs, particuliers et professionnels, d'accéder aux services composés et aux informations souhaitées n'importe quand et n'importe où en utilisant un terminal (PDA, GSM, Tablette, etc.). Un large éventail des LBS fonctionne d'une façon individuelle et indépendante et sans aucun partage d'information. Pour remédier à ce problème, la plateforme propose une solution d'interopérabilité et d'interaction de services permettant d'améliorer la qualité des services, réduire les coûts, réduire les temps de réponse, etc. A titre d'exemple, un service d'urgence sollicité pour porter secours à une personne en situation de malaise doit se renseigner sur des informations telles que le plus court chemin amenant vers l'emplacement du patient, l'état des routes, les routes déneigées, etc. L'accès à toute information pertinente de ce type permet d'accélérer l'opération de secours et fournir une assistance plus sûre et plus rapide.

Notre plateforme de collaboration et coopération se base sur une architecture générale que nous développons dans ce qui suit. Cette architecture est la base des différents scénarios développés. Un scénario global impliquant plusieurs services métiers de la vie courante, tels que le service de déneigement, le service d'état des routes, le service de la police, le service d'urgence, le service des hôpitaux, le service Geofencing, est proposé pour illustrer notre approche. Afin d'étudier la faisabilité du scénario et d'évaluer ses propriétés de bon fonctionnement, nous représentons son comportement graphique et analytique par un modèle RdP et par des équations mathématique dans l'algèbre (max,+). Après cette phase de spécification et de modélisation, nous implémentons le scénario et rapportons des résultats des tests et des expérimentations menés sur le terrain. Une étude comparative des résultats obtenus par des modèles formels et ceux des tests réels est exposée à la fin du chapitre.

2 Architectures générales

Nous présentons dans cette section, l'architecture générale adoptée pour tous les scénarios développés.

2.1 Architecture générale des scénarios

L'architecture générale de tous les scénarios implémentés dans un serveur, appelé par la suite serveur ASSET, est présentée dans la figure V.1. Cette architecture est basée principalement sur des technologies de communication et des systèmes de géolocalisation. Ces technologies permettent d'une part la communication des applications avec le serveur via Internet, et d'autre part la localisation de véhicules en temps réel.

Cette architecture est la base de tous les scénarios décrits dans les sections suivantes. L'interaction entre tous les composants de l'architecture se fait selon les principes suivants : positionnement en temps réel en utilisant les services de positionnement et de navigation par satellite (GPS et EGNOS), communication bidirectionnelle entre les véhicules et l'infrastructure (V2I2V) via les technologies sans fils (GPRS ou 3G), accès aux informations et aux données via des bases de données et des systèmes d'information, sécurité des réseaux (contrôle d'accès...). Ces principes sont détaillés dans ce qui suit.

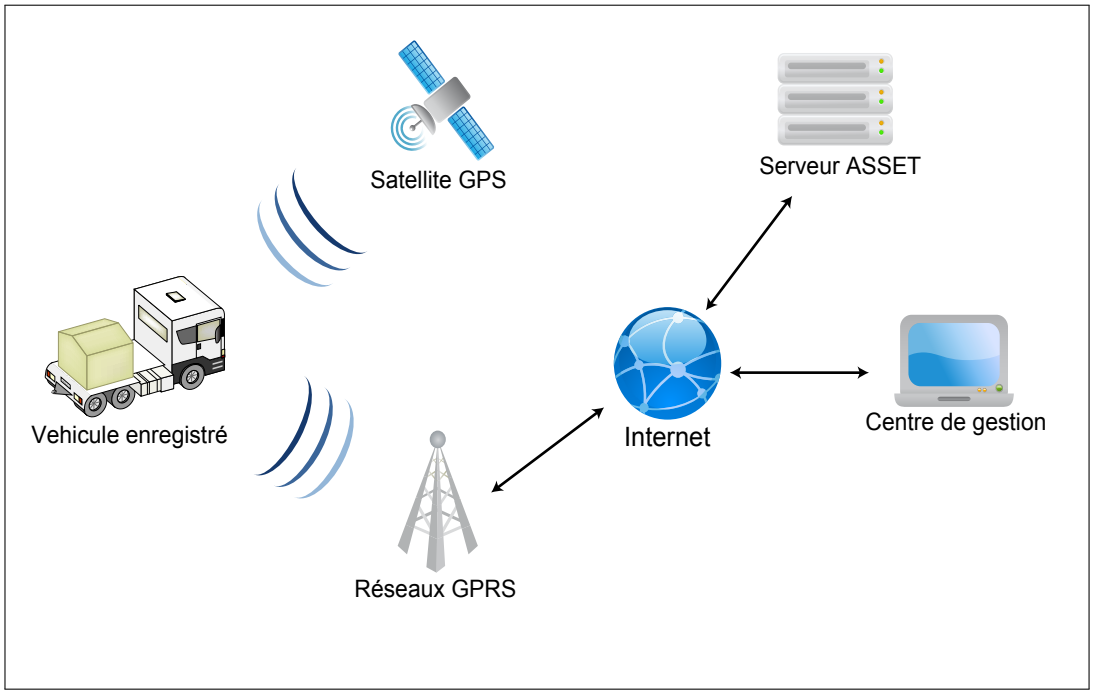


Figure V.1 – Architecture générale des scénarios

2.2 Architecture de communication V2I2V

TransportML comprend une couche de communication basée sur la technologie GPRS pour permettre la communication entre les véhicules et la plateforme. Il s'agit des communications véhicule-infrastructure et infrastructure-véhicule (V2I et I2V). La plateforme TransportML n'est pas en mesure d'initier les connexions vers les véhicules identifiés par des adresses IP privées cachées derrière un serveur NAT [3] (Network Address Translation), d'où la nécessite du développement et de l'intégration de cette couche de communication. Il est important de signaler que TransportML ne peut pas identifier tous les véhicules enregistrés, et que la communication directe V2V n'est pas supportée du fait des adresses IP privées des véhicules. De même, d'une part un serveur NAT utilisé pour accéder à Internet par la technologie 3G lui impose une adresse IP publique. D'autre part, la communication mobile utilisant le réseau 3G est peu fiable en raison des déconnexions fréquentes, d'une latence élevée et d'un faible débit.

Pour permettre la communication V2I ou I2V entre les applications embarquées dans les véhicules et l'infrastructure (c.-à-d. serveur de traitement), nous proposons une nouvelle architecture de communication. Cette dernière est présentée dans la figure V.2. Elle est composée

3. NAT est utilisé pour faire face aux adresses publique et privée. En effet, la plupart des adresses que nous utilisons sont privées et afin d'utiliser Internet, il nous faut des adresses publiques. Le serveur NAT est donc nécessaire pour la conversion d'adresses.

de la librairie *LibMobileCom* et du serveur de relais *RelayServer*.

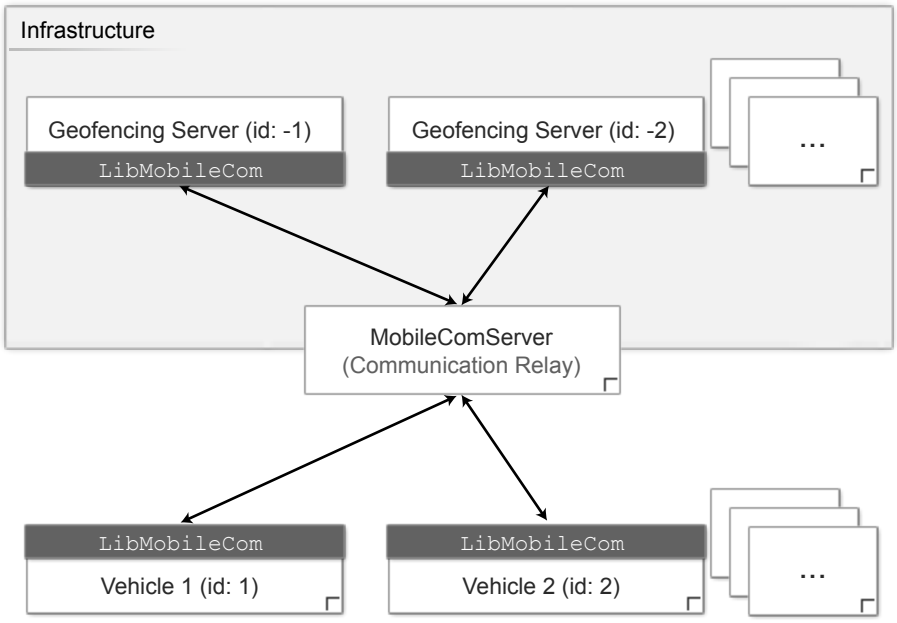


Figure V.2 – Architecture de la communication V2I/I2V

Il est à noter que chaque entité, faisant partie de cette architecture (figure V.2), possède un identifiant unique. Par convention, nous utilisons des identifiants de valeurs négatives pour toutes les entités appartenant à l'infrastructure et de valeurs positives pour les autres entités (p. ex. véhicules). Pour se connecter au serveur relais et échanger des messages, toutes les entités utilisent la librairie LibMobileCom.

2.2.1 LibMobileCom

La librairie LibMobileCom [4], développée en JAVA, utilise les sockets TCP pour établir la communication entre les différentes entités. Elle est capable de gérer différente tâche à savoir, l'établissement de la communication, la détection de la déconnexion, la reconnexion automatique. Elle s'occupe également de formatter les messages d'échange avec l'infrastructure. Ce format est décrit dans la figure V.3. La librairie gère également le buffering des messages avec une priorité dans le cas d'indisponibilité du réseau et l'acheminement de ces messages après l'établissement du réseau.

4. http ://www.gsem.fr/V2V/

2.2.2 RelayServer

Le serveur relais *RelayServer* est un serveur de communication, qui joue le rôle d'un relayeur de messages. Il est connecté à Internet via une adresse IP publique pour gérer toutes les communications entre les entités impliquées. L'objectif de ce serveur est de maintenir le canal de communication ouvert à tout moment entre les véhicules, la plateforme TransportML, et les applications dans l'infrastructure. Toutes les entités qui communiquent via l'architecture, envoient et reçoivent des messages via ce serveur central. Le serveur est considéré donc comme un récepteur et émetteur des messages vers leurs destinataires. Pour ce faire, le serveur relais analyse uniquement l'ID du destinataire et le message prioritaire, puis transmet le message. Si le destinataire n'est pas disponible et que le message est prioritaire, ce dernier sera stocké dans une file de type FIFO (First In First Out). Sinon, il sera immédiatement transmis une fois qu'il a été analysé par LibMobileCom. Une description du message échangé est donnée dans la section suivante.

2.2.3 Message échangé

Le format du message échangé entre les différentes entités qui interagissent via le serveur relais, est illustré par la figure V.3. Dans cette figure, nous distinguons six champs décrits ci-après.

Figure V.3 – Le format du message échangé

1. *Type du message* : correspond au type du message échangé et donc définit le type de la requête souhaitée,
2. *ID Destinataire* : identifie l'ID du destinataire de la requête,
3. *ID Emetteur* : sert à identifier l'expéditeur du message,
4. *Priorité* : ayant une valeur booléenne permettant de définir les messages avec une priorité et donc les messages qui ne seront pas supprimés quand le destinataire ou le réseau est non disponible,
5. *Timestamp* : définit la date d'envoi du message,
6. *Données* : c'est le contenu du message qui est interprété en fonction du type du message.

Le Tableau V.1 décrit les différents types de message supportés par notre environnement. Il s'agit de différents messages échangés d'une façon unidirectionnelle ou bidirectionnelle, selon les applications, entre les véhicules et les différentes applications de l'infrastructure.

Tableau V.1 – Les types de message supportés

Type de message	Description
GPS	Position GPS
TML	Requête / Réponse TransportML
EFA	Entrée dans une zone interdite
LFA	Sortie d'une zone interdite
ABS	Déclenchement de l'ABS
WRN	Information sur le déclenchement de l'ABS
WLE	Dépassement de la limite en poids
HLE	Dépassement de la limite en hauteur
SLE	Dépassement de la vitesse autorisée

Le contenu d'un message pourrait changer en fonction des types décrits ci-dessus. Par exemple, lorsque le message est de type GPS, le contenu est de format : *<Latitude ; Longitude>* (en décimal). Pour d'autres types, le contenu pourrait être vide. C'est le cas des messages de type : HLE ou SLE où on signale un dépassement de poids ou de vitesse sans vouloir envoyer une information précise.

Il est intéressant de noter que, cette solution prend en compte la déconnexion des entités en enregistrant, en fonction de leurs priorités, les messages à envoyer dans le cas de non disponibilité du réseau ou du destinataire. Seuls les messages ayant une faible priorité sont supprimés afin d'éviter tout débordement de la mémoire tampon. En outre, la surcharge du protocole est maintenu au minimum afin d'économiser la bande passante et de diminuer la latence.

3 Scénarios réalisés

Tous nos scénarios ont été développés en langage de programmation *Java/J2EE*. Nous avons utilisé l'API libre *GoogleMaps* de *Google* pour représenter les données sur la carte, ainsi que les langages Web *Javascript* et *AJAX* (p. ex. *Mootools*[5] et *jQuery*[6]) pour le rafraîchissement de la carte et la collecte des informations en dur depuis les serveurs. Le système de gestion de base de données MySQL est utilisé pour enregistrer certaines informations sur les zones

5. La librairie Mootools, http ://mootools.net/
6. La librairie jQuery, http ://jquery.com/

géographiques et leurs caractéristiques. Nous utilisons également dans certains scénarios des fichiers XML contenant les informations sur les itinéraires et les routes.

Nous utilisons également une API que nous avons développé pour récupérer les trames GPS depuis un dispositif connecté à l'ordinateur via le port série. Cette API définit toutes les méthodes et fonctions de lecture, de conversion des coordonnées (c.-à-d. conversion du format géodésique WSG84 au format décimal de la latitude et de la longitude), de filtrage d'informations issues en lecture de l'appareil. Il est important de savoir, qu'il existe plusieurs types de trames que l'on peut récupérer depuis le dispositif GPS (p. ex. GGA pour Global Positioning System Fix Data ou WPL pour *Waypoint* location), nous nous sommes limités à la trame *RMC*[7] (Recommended minimum data) qui contient les informations nécessaires pour notre développement à savoir, l'attitude, la longitude, la vitesse et le temps.

3.1 Scénario de Geofencing

Le terme Geofence est composé de deux mots : «fence» qui signifie une clôture ou un périmètre et «Geo» qui signifie que ce périmètre est construit en se basant sur des données géographiques. La notion de Geofencing, ou Geocorridor, est utilisée par plusieurs applications basées sur les systèmes GNSS. Ces applications ont divers objectifs en fonction de leur domaine métier. Nous en citons quelques exemples, tels que les applications de gestion du transport des marchandises dangereuses (TDG pour *Transportation of Dangerous Goods*), les applications permettant aux compagnies d'assurance de gérer les zones où leurs clients sont assurés [tea10].

Historiquement, le principe du Geofencing a été utilisé par les entreprises pour limiter ou surveiller les déplacements de leurs employés dans une zone géographique définie comme leur domaine de travail [REC09], l'application permettra ainsi de s'assurer que les véhicules de l'entreprise circulent dans un périmètre autorisé.

Dans la pratique, le Geofencing permet de définir un périmètre virtuel autour d'une zone géographique, puis en associant des objets (p. ex. véhicule, personne, etc.) à cette zone, une alarme est émise lorsque l'objet traverse la frontière du périmètre.

La figure V.4 illustre un exemple d'un Geofence. En effet, la ligne rouge définit la frontière d'une zone surveillée interdite aux véhicules non autorisés. Le système Geofencing est appliqué à cette zone dans le but de détecter les véhicules qui pénètrent dans la zone. Afin de surveiller la zone, les positions (déterminées par les coordonnées GPS) de tous les véhicules circulant à proximité de ses frontières sont identifiés par le système. En outre, la position en temps réel de chacun de ces véhicules est envoyée au système et affichée sur une carte numérique. Lorsqu'un

7. Les trames NMEA, http ://www.gpsinformation.org/dale/nmea.htm

Figure V.4 – Exemple de Geofencing

véhicule franchit la ligne rouge, il est détecté automatiquement et une alerte est déclenchée. Cette alerte est envoyée au conducteur pour l'informer de son intrusion dans une zone interdite. Dans certains cas, l'alerte est envoyée également à d'autres acteurs intéressés, par exemple dans le cas du transport de marchandises dangereuses, l'alerte est envoyée au gestionnaire de flotte afin de l'informer au plutôt de la situation.

Nous avons développé trois applications de Geofencing. La première application consiste à gérer des zones géographiques en précisant les caractéristiques de chaque zone. La deuxième application porte sur le suivi et le contrôle d'accès des véhicules aux zones prédéfinies, ainsi que l'envoie d'alertes lorsqu'un véhicule franchit la frontière d'une zone. Dans la dernière application il s'agit d'un service à base de localisation, appelé aussi Geofencing, permettant de fournir les zones interdites par rapport à un profil de véhicule. Ce service est utilisé par la plateforme TransportML pour le calcul d'itinéraires en évitant les zones interdites ou déconseillées pour un type donné de véhicules. Ces applications sont exposées en détails dans ce qui suit.

3.1.1 Application de définition des Geofences

La figure V.5 illustre l'interface web que nous avons développé pour définir un Geofence et ses caractéristiques. Afin de définir un Geofence, les caractéristiques de la zone doivent être renseignées. Il s'agit, par exemple, des points géographiques sur la carte (c.-à-d. les marqueurs en rouge au contour de la zone constituent les points définissant le périmètre de la zone), le type des véhicules interdits (c.-à-d. poids lourd, voiture ou moto), la vitesse limitée dans la zone, la hauteur limitée et le poids des véhicules limité. Toutes les zones et leurs caractéristiques sont

enregistrées dans une base de données. Un exemple de scénario est donné ci-après.

Un poids lourd désirant se déplacer d'un point de départ vers un point d'arrivée doit être informé des zones et des itinéraires interdits lors de son déplacement. En effet, le poids lourd empruntant, sans être informé, un itinéraire avec un passage restrictif (p. ex. pont à hauteur limitée) sans possibilité de traverser ni de faire un demi-tour se verra dans l'obligation de s'arrêter avant d'être conduit et guidé par des acteurs de sécurité (p. ex. police, services des routes) ce qui a un impact important sur le coût, le temps, le trafic, etc. Des systèmes de sécurité, tels que le Geofencing, proposent donc des solutions pertinentes pour remédier à ce type de problème et augmenter le niveau de la sécurité dans les transports.

Figure V.5 – Imprimé écran de l'interface de définition de Geofence

Par ailleurs, nous avons ajouté également le paramètre temporel associé aux zones, permettant ainsi d'activer ou de désactiver un Geofence en fonction d'une date ou d'une heure déterminée. Par exemple, si nous considérons une école comme un Geofence, l'accès des véhicules à cette zone sera alors interdit dans la semaine du lundi au vendredi et entre sept heure du matin jusqu'au dix neuf heure du soir comme décrit dans la figure V.6.

Comme toutes les zones sont enregistrées dans une base de données, il est possible de mettre à jour les Geofences en fonction des besoins des intéressés. Cette base de données ainsi que les Geofences qui peuvent être générés sont accessibles et utilisés par les deux autres applications décrites dans les sous-sections suivantes.

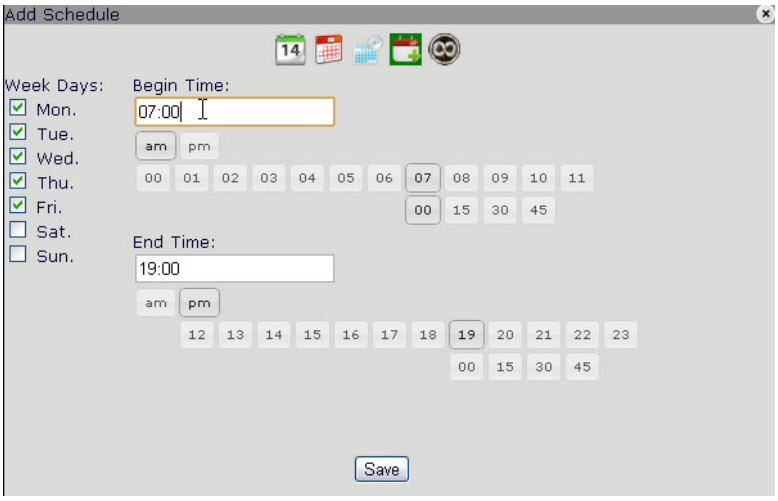


Figure V.6 – Imprimé écran de l'interface de définition de la date et l'heure d'activation d'une zone

3.1.2 Application du contrôle Geofencing

L'application de contrôle a pour rôle de suivre et de surveiller les déplacements de tous les véhicules enregistrés. La communication bidirectionnelle entre le serveur et les véhicules se base sur la même architecture décrite précédemment (figure V.2). En effet, cette application implémente la librairie *LibMobileCom* et utilise un ID pour se connecter au *RelayServer* afin d'envoyer et de recevoir les messages via ce serveur. Les différents types de messages supportés par cette application sont EFA, LFA, WLE, SLE et GPS (Tableau V.1).

L'application de contrôle reçoit constamment les messages GPS issues des véhicules. Le système récupère les coordonnées GPS et représente les positions des véhicules sur la carte. Il identifie ensuite depuis la base de données les zones interdites, qui sont également représentées sur la carte, pour chaque type de véhicule communiquant ses coordonnées GPS. Ensuite, il vérifie si la position du véhicule est à l'intérieur ou à l'extérieur des zones interdites. Pour ce faire, il calcule la position GPS par rapport aux points caractéristiques de chaque zone en s'appuyant sur un algorithme géométrique implémenté en JAVA (Classe Polygon). Dans le cas où le véhicule se trouve à l'intérieur d'une zone interdite, le système envoi un message EFA pour informer le conducteur du véhicule qu'il se trouve dans une zone interdite. La figure V.7 représente le message reçu par le système Geofencing via le RelayServer, et envoyé au véhicule. Dès que ce dernier sort de la zone, un message LFA (figure V.8) est envoyé via le système et reçu par le véhicule.

En même temps que la position GPS, le système enregistre toutes les informations, envoyées par le véhicule, relatives à son poids, ses dimensions, à la matière transportée (pour les poids lords), etc. Le contrôle se fait également en fonction de ces paramètres. La vitesse en temps

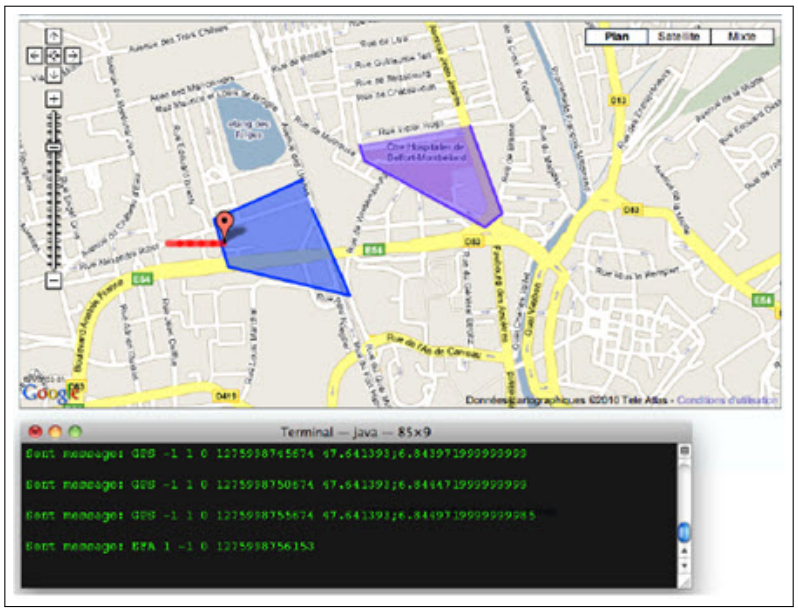


Figure V.7 – Véhicule entrant dans une zone interdite

réel est récupérée depuis le dispositif GPS embarqué dans le véhicule. Lors de son déplacement, le conducteur peut être informé, par des messages de type SLE, de tout excès de vitesse. Dans les tests et expérimentations réels que nous avons effectués, nous avons évalué certaines performances du système en termes de latence et de la distance parcourue par le véhicule à partir du moment où il franchi la frontière de la zone et le moment où il reçoit le message EFA (resp. LFA). Les résultats obtenus montrent l'efficacité de l'application et la rapidité dans la transmission et la réception des messages.

3.1.3 Le service Geofencing

La troisième application développée est le service à base de localisation *Geofencing*. C'est un service Web utilisé par la plateforme TransportML pour le calcul d'itinéraires en évitant des zones interdites par rapport à des types de véhicules spécifiques. En effet, lors de l'envoie d'une requête, l'utilisateur spécifie en plus de ses points de départ et d'arrivée, les caractéristiques de son véhicule. Cependant, quand TransportML envoie le TMLDocument (document échangé entre la plateforme TransportML et les différents LBS à faire collaborer) à ce service, les sections *route*, *waypoints* et *vehicle* (voir chapitre 4, §2.2) sont renseignées dans le document. Compte tenu de ces informations, le service va tout d'abord sélectionner les zones interdites à un type de véhicule spécifié puis remplira la section correspondante au champ *unadvised_areas*. Par la suite l'itinéraire sera calculé en tenant compte de ces informations. En d'autres termes, ce service

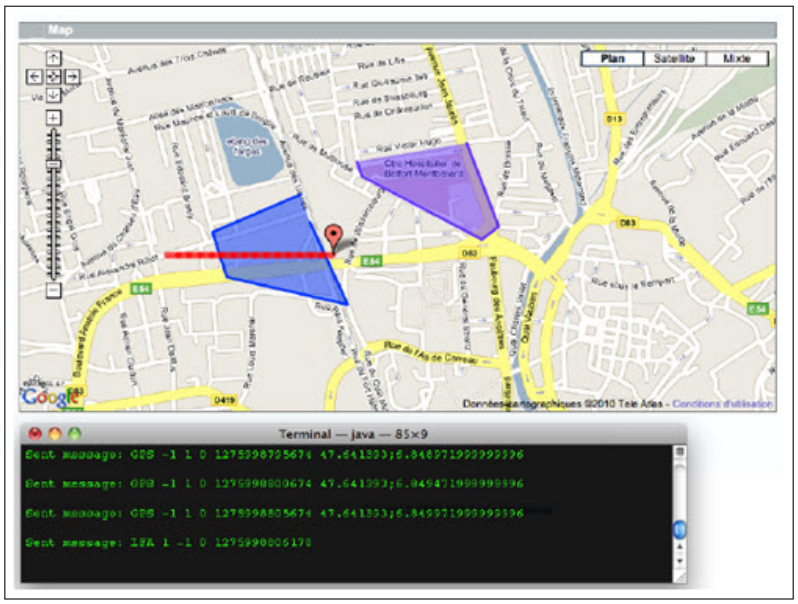


Figure V.8 – Véhicule sortant d'une zone interdite

fournit les informations sur les zones autorisées ou interdites aux véhicules en fonction de leurs caractéristiques. De même, il fournit les informations concernant les zones où la circulation peut être difficile et où la congestion peut apparaître.

3.2 Scénario de déneigement

Nous avons réalisé deux applications pour le déneigement. La première application consiste à représenter en temps réel les routes déneigées et non-déneigées ainsi que le déplacement en temps réels des déneigeuses. La deuxième application porte sur la mise en place d'un service qui permet de fournir l'état des routes en temps réel (c.-à-d. routes non déneigées). Ces applications sont décrites dans ce qui suit.

3.2.1 L'application de déneigement

L'application de déneigement permet de surveiller et de suivre en temps réel les déplacements des chasse-neiges et l'état d'avancement du processus de déneigement. L'objectif de cette application est de visualiser le déplacement de chacune des chasse-neiges enregistrées en communicant avec le serveur. Au même titre que d'autres applications développées dans ce travail, cette application se base sur l'architecture de communication décrite auparavant. Les itinéraires des chasse-neiges ainsi que l'historique de leurs déplacements sont stockés dans des fichiers XML.

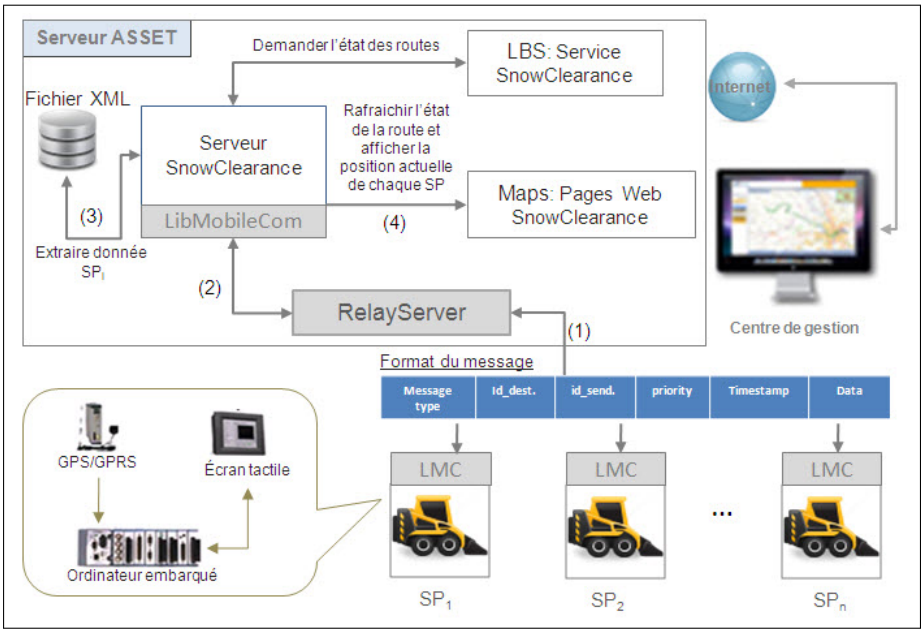


Figure V.9 – Architecture de l'application déneigement

L'architecture de cette application est présentée dans la figure V.9. Comme le montre cette figure, chaque chasse-neige (SP) est équipé d'un ordinateur embarqué, un récepteur GPS et un module de communication GPRS pour la transmission de données. Quand un SP envoie un message de type GPS, le système analyse d'abord le message reçu pour obtenir les coordonnées GPS et l'ID du chasse-neige. Ensuite, il vérifie dans le fichier XML les itinéraires assignés au chasse-neige, et enfin met à jour les routes déneigées dans le fichier XML et rafraîchit la carte.

La figure V.10 représente l'activité du déneigement par quelques déneigeuses. Les lignes en rouge représentent les portions non déneigées de la route et les lignes en vert représentent les portions déneigées. L'emplacement et le déplacement des déneigeuses sur les routes sont affichés en temps réel sur la carte, et le rafraîchissement de la carte se fait régulièrement toutes les 3 secondes.

Afin d'illustrer et de valider l'application développée, certaines expériences et tests ont été menés dans un environnement réel sur le terrain. Ces tests ont montré que l'automatisation du processus de déneigement est essentielle, d'une part, pour le personnel afin d'effectuer leur tâche d'une façon efficace, et d'autre part, pour les gestionnaires afin de suivre et surveiller leurs véhicules de déneigement et de prendre les bonnes décisions en cas d'incidents. Les informations enregistrées sont partagées avec d'autres services via la plateforme TransportML afin de permettre aux utilisateurs, y compris le grand public, de la route d'éviter les routes non déneigées dans leurs déplacements.

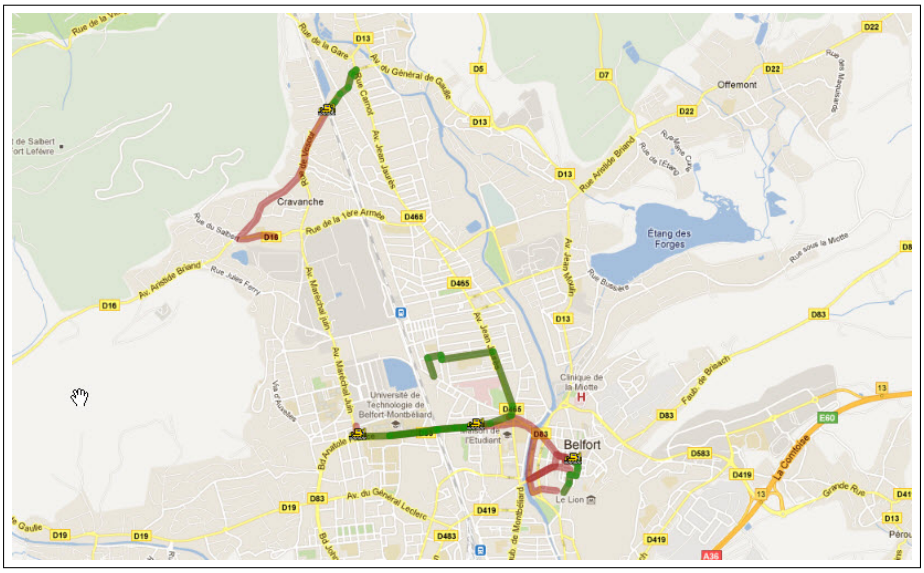


Figure V.10 – Statut des itinéraires au moment du déneigement

Le service de déneigement fournit le statut des routes en temps réel. Il se base sur le fichier XML contenant les itinéraires et qui est mis à jour chaque fois que le chasse-neige envoie l'information au système. Ce service est également utilisé par la plateforme TransportML afin d'éviter les routes non déneigées lors du calcul d'itinéraires suite à une requête d'un utilisateur.

3.3 Scénario Anti-Blocage de Sécurité (ABS)

Le but de ce scénario est de développer une application côté serveur qui échange des informations liées au déclenchement de l'ABS entre les véhicules. Ces informations sont transmises en précisant les coordonnées GPS du lieu de déclenchement de l'ABS via une connexion 3G. Le véhicule envoie ces informations au serveur, qui à son tour les relaye aux véhicules situés à proximité du lieu de l'enclenchement de l'ABS. Un module a été développé, par notre équipe, en Labview pour récupérer les informations liées à l'ABS depuis le bus CAN des véhicules [Cai11]. Ce même module permet également la communication avec le serveur relais pour transférer ces informations. Pour pouvoir traiter les informations reçues, nous avons développé un serveur *ABSServer*. Ce dernier utilise la librairie *LibMobileCom* pour communiquer avec les différentes entités. Chaque véhicule envoie des messages "GPS" au serveur, qui retient régulièrement les deux dernières positions des véhicules. Quand le serveur reçoit un événement ABS, il commence par filtrer la liste des véhicules pour déterminer ceux qui se trouvent à proximité de cet événement. Finalement, un message d'alerte "déclenchement ABS" ou "route glissante" est envoyé à ces véhicules. Pour les expérimentations de ce scénario, un périmètre de 2.2 km sur

1.6 km au tour du lieu de l'enclenchement de l'ABS a été considéré et tout véhicule, équipé de l'application développée, se trouvant à l'intérieur de ce périmètre est informé de l'évènement.

La librairie *LibMobileCom* a été implémentée en *LabView* et intégrée dans l'ordinateur embarqué du véhicule pour permettre les échanges d'information V2I2V.

3.4 Scénario de TransportML

Le scénario basé sur l'utilisation de la plateforme TransportML a été développé en considérant tous les services définis dans les sous-sections précédentes. En effet, nous avons considéré, comme scénario global pour valider notre approche, le processus de calcul d'itinéraire en évitant certaines zones géographiques et les routes déconseillées fournies par le service de déneigement, par le service de travaux publics et par le service de Geofencing. Nous décrivons dans les deux sous-sections suivantes, les autres services impliqués dans ce calcul d'itinéraire. Il s'agit du service itinéraire, basé sur un algorithme d'optimisation (Dijkstra) pour le calcul du plus court chemin, et le service états des routes (ou travaux publics) qui fournit en temps réels les informations liées à la présence des travaux en cours sur les routes.

3.4.1 Service travaux publics

Le service à base de localisation travaux publics fournit régulièrement et en temps réel la liste des zones et routes en cours de travaux. Ainsi, il renseigne les deux sections dans le document TML à savoir, la section *<advised_area>* pour les routes n'étant en travaux et la section *<unadvised_area>* pour les zones en travaux et qui ne sont pas conseillées aux conducteurs. Ce service pourrait, par exemple, être fourni par les autorités et les collectivités locales. Nous signalons que les données issues de ce service sont définit d'une façon statique.

3.4.2 Service itinéraire

Le service itinéraire permet de calculer un itinéraire optimal entre un point de départ et un point d'arrivée. En effet, ce service est basé sur l'algorithme d'optimisation Dijkstra pour le calcul du plus court chemin en considérant les tronçons d'une route sous forme d'un graphe. Pour ce faire, nous utilisons les fichiers de données géographiques *Shapefile* d'extension ".shp" qui permettent de définir le format des objets (p. ex. polygone, ligne, point), les fichiers d'index du format d'extension ".shx" qui sont utilisés pour faciliter la recherche des formes, et les attributs de chaque forme qui sont stockés dans des fichiers d'extension ".dbf".

Nous avons abordé dans ce qui précède les différentes applications et scénarios réalisés dans

le cadre des projets Européens ASSET et TeleFOT. Certaines de ces applications participent indirectement dans l'élaboration du scénario considéré dans l'étude de cas, développés pour approuver nos approches formelle et applicative. Des expérimentations ont été réalisées pour évaluer les performances de ces applications sans donner plus de détails, c'était le cas des applications Geofencing, ABS et déneigement. Nous avons montré quelques aboutissements sous forme de copies d'écran des résultats obtenus. D'autres applications ne sont pas détaillées dans ce mémoire, par exemple la mise en place du service e-Call où plus de détail est donné dans [ACBCB+11].

4 Étude de cas

Dans cette section, nous allons étudier un scénario d'urgence élaboré dans le cadre des projets ASSET et TeleFOT dans lesquels, les applications et les services définis précédemment sont impliqués. Ce scénario a pour but de montrer la faisabilité et l'apport de la plateforme TransportML ainsi que d'illustrer son impact sur la coordination distribuée des LBS. Le scénario est basé sur des informations liées à la route et recueillies auprès de différents services locaux, c.-à-d. état des routes, déneigement et Geofencing. Dans ce scénario nous nous intéressons au calcul d'un itinéraire optimal. Les services d'urgence font appel à ce processus de calcul d'itinéraire afin de joindre le lieu d'un incident le plus rapidement possible, et en évitant certaines zones (p. ex. routes non déneigées) pouvant ralentir ou perturber le déplacement des véhicules d'urgence.

Nous allons commencer par décrire en détail les différents acteurs et services participants à ce scénario. Ensuite, nous allons le modéliser en utilisant les deux outils formels retenus pour ce travail à savoir les RdP et l'algèbre (max,+). Dans un premier temps, le scénario sera représenté sous forme d'agrégation de workflow patterns vus dans le chapitre 3. Ensuite chaque pattern sera formalisé par des équations mathématiques. La méthodologie se généralisera enfin pour déterminer le modèle global du scénario.

4.1 Description

L'objectif de ce scénario est de minimiser le temps d'intervention des acteurs de sécurité (service d'urgence, autorités, etc.) en cas d'incident d'urgence (accident, malaise, crise, etc.). Pour cela, les autorités devront contacter la plateforme TransportML afin de demander le calcul d'un itinéraire optimal vers le lieu de l'incident. Ceci est réalisé à travers l'invocation de services permettant de fournir des informations importantes sur les états des routes. L'interopérabilité et la collaboration entre ces services représente une valeur ajoutée produite par la plateforme

TransportML.

Nous disposons de quatre services Web liés aux urgences utilisés afin d'intervenir en cas d'incidents dangereux. Lorsqu'un conducteur ou utilisateur signale un incident, la collaboration et l'interaction entre ces services permettent de lui fournir des informations nécessaires pour son assistance telles que la confirmation d'intervention des services concernés, une estimation du temps d'intervention de chaque service, etc. Les figures V.11 montrent les différentes interactions entre ces services et la plateforme TransportML.

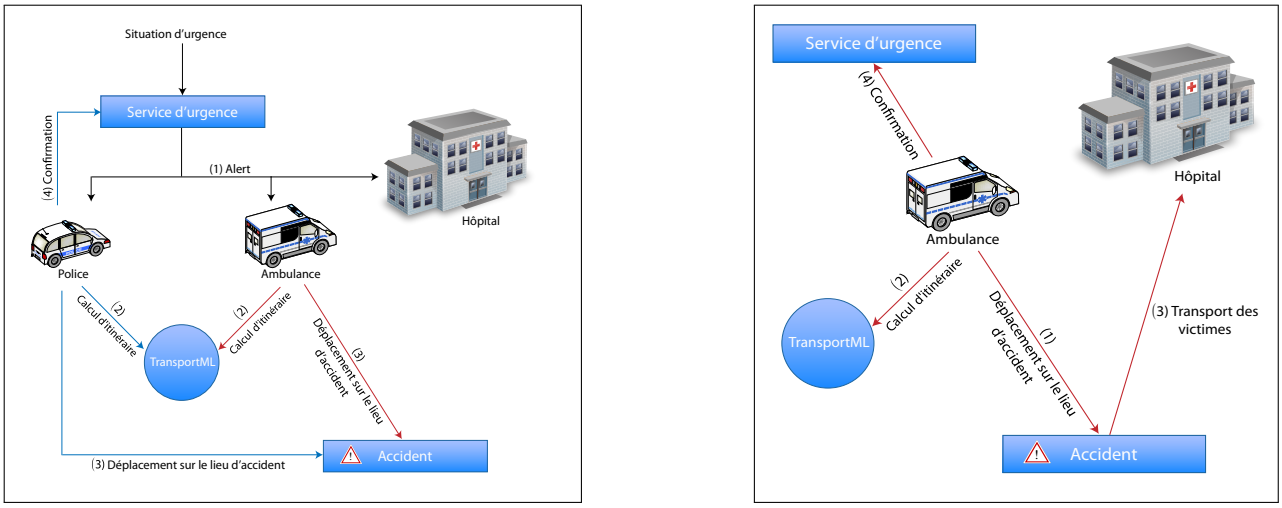


Figure V.11 – Description de l'étude de cas

Comme illustré dans la figure V.11, le prototype consiste à réaliser un processus permettant l'intervention des services de secours d'urgence suite à un accident sur la route. Lorsqu'une personne subit un accident, elle contacte le service d'urgence (p. ex. appeler le numéro 112) [ACBCB+11], ce dernier déclenche trois processus en parallèle. Le premier consiste à avertir les ambulanciers en leur rapportant le lieu de l'accident. Cela se fait via l'invocation du *service ambulance* qui permet de contacter le centre d'ambulances le plus proche du lieu de l'accident. Le deuxième processus, consiste à informer la police en appelant le *service police* qui contacte le poste de la police le plus proche du lieu de l'accident. Le dernier processus consiste à contacter le *service hôpital* pour chercher l'hôpital le plus proche du lieu de l'accident. Ce premier scénario est décrit dans la figure V.11a.

Les services *ambulance* et *police* vont donc demander le calcul d'itinéraire à partir de leur position respective de départ (coordonnées GPS) vers l'endroit de l'accident. Pour cela, ils envoient des requêtes à la plateforme TransportML afin de leur fournir l'itinéraire en évitant

les zones et les routes interdites ou déconseillées. Pour transporter les blessés vers l'hôpital, le service *hôpital* fournit l'adresse de l'hôpital le plus proche et ayant la capacité de recevoir les patients. Comme décrit dans la figure V.11b, le service *ambulance* va donc demander à recalculer, via TransportML, un nouveau itinéraire optimal à partir du lieu de l'accident vers l'hôpital.

Nous résumons dans ce qui suit les fonctionnalités fournies par chaque service :

- Le service *Permanence* est appelé par l'utilisateur depuis son appareil embarqué dans le véhicule (GSM, PDA, etc.) [ACBCB+11] en fournissant de façon automatique ses coordonnées de localisation et le type de l'incident. Il redirige ensuite la demande aux services concernées, tels que la *Police* et/ou *Hôpital* et *Ambulance* en fonction de la nature de l'incident (p. ex. dans le cas d'un accident les trois services *Police*, *Ambulance* et *Hôpital* sont invoqués).
- Le service *Police* reçoit la requête du *service Permanence* avec la position de l'incident. Il contacte ensuite le poste de police le plus proche à l'incident. Ce service utilise la plateforme TransportML [ACBBNSM+11] pour calculer l'itinéraire de sa position actuelle vers le lieu de l'incident.
- Le service *Ambulance*, comme le *service Police*, reçoit la requête avec le lieu de l'incident et contacte ensuite le service d'ambulance le plus proche à ce lieu. L'itinéraire sera ensuite fournie par la plateforme TransportML en lui faisant la demande.
- Le service *Hôpital* reçoit le lieu de l'incident et fournie au *service ambulance* l'hôpital le plus proche. Un autre itinéraire sera calculé par le service *ambulance* du lieu de l'incident vers l'hôpital.

Ce scénario sera modélisé par les workflow patterns et sera étudié de façon formelle dans la section suivante.

4.2 Modélisation et étude de performances du prototype

Ce prototype est modélisé et évalué par notre approche en se basant sur les patterns définis précédemment dans le chapitre 3. Dans cette section, nous détaillons la description de ce scénario, puis nous représentons son modèle RdP à partir de l'agrégation des modèles RdP des différents patterns dont il est composé. Enfin, nous décrivons son comportement analytique par des équations d'état dans l'algèbre (max,+).

Une étude de performances est réalisée en utilisant les équations (max,+) obtenues. Il est important de noter que, lors de la modélisation, divers critères pour la composition dynamique de services Web sont pris en compte et sont considérés par la suite lors de la phase de validation.

Le modèle RdP du scénario décrit dans la section précédente est illustré par la figure V.12. Il est composé des patterns suivants : sequence (x_4, x_7 et x_{10}), multi-fusion (x_1, x_2 et x_3), choix différé (U, x_1 et x_2), synchroniseur (x_4, x_5 et x_8), et fractionnement parallèle (x_2, x_3, x_4 et x_5). Les places compteurs Q_1, Q_2, Q_3 et Q_4 sont ajoutées comme décrit dans la section 3.4 pour faciliter la mise en équation. Nous rappelons que ces places ne participent en aucun cas à l'évaluation et à l'analyse du modèle. Le tableau V.2 décrit la signification des différents éléments du modèle RdP.

Tableau V.2 – Légende du modèle RdP de la figure V.12

Transition/Place	Signification
U	Recevoir une requête d'urgence
P_1	WS Permanence
x_1	Informer le service Police dans le cas d'un incident non dangereux
x_2	Informer tous les services dans le cas d'un incident dangereux
P_2	Recherche du poste de police le plus proche
x_3	Fournir la position du poste de police
P_3	Recherche du poste d'urgence le plus proche
x_4	Fournir la position du poste d'urgence
P_4	Recherche de l'hôpital le plus proche
x_5	Fournir la position de l'hôpital
P_5	Demande d'itinéraire de la position de départ des agents de la police vers la position de l'incident
x_6	Fournir l'itinéraire demandé
P_6	Demande d'itinéraire de la position de départ de l'ambulance vers le lieu de l'incident
x_7	Fournir l'itinéraire demandé
P_7	Attente de la réponse de l'ambulance
P_8	Attente de la réponse de l'hôpital
x_8	Attente des deux réponses des service ambulance et hôpital
P_9	Le temps nécessaire pour le déplacement de la police vers le lieu de l'incident
P_{10}	Le temps nécessaire pour le déplacement de l'ambulance vers le lieu de l'incident
P_{11}	Demande d'itinéraire de la position de l'incident à l'hôpital par le service d'urgence
x_9	Fournir l'itinéraire demandé
P_{12}	Le temps de parcours de l'ambulance du lieu de l'incident à l'hôpital
x_{10}	Fournir toutes les réponses pour le calcul du temps d'intervention global
x_{11}	Fournir la réponse du service police
P_{13}	La réponse finale de la requête

Comme expliqué précédemment dans le chapitre III, la sélection de services se fait en fonction des critères de la qualité de chaque service. Compte tenu de ces critères de la QoS, les temporisations associées aux places du modèle RdP correspondent à celles des meilleurs services sélectionnés. Nous soulignons également que les places P_5, P_6 et P_{11} représentent le processus du calcul d'itinéraire pour chaque service impliqué. En réalité, la représentation de ce processus

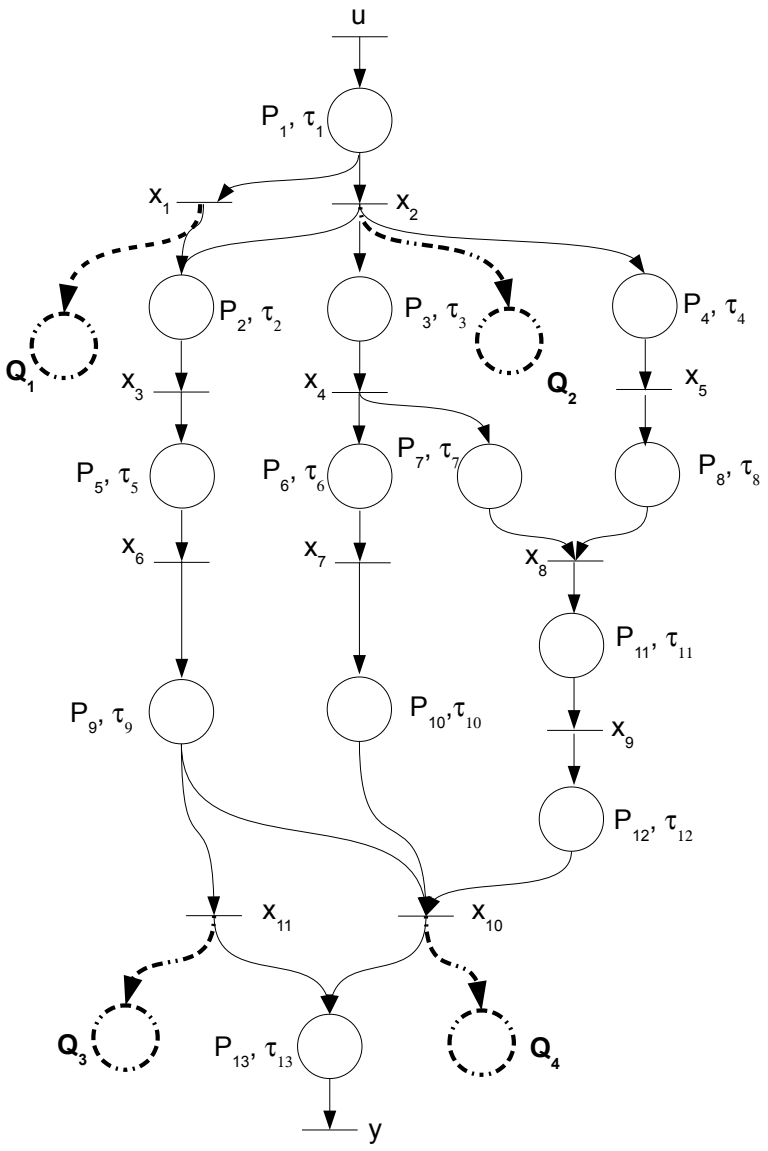


Figure V.12 – Le modèle RdP du scénario

par ces places permet de diminuer la complexité spatiale du modèle RdP et de réduire la taille des matrices caractéristiques du modèle mathématique associé. Le modèle RdP de ces places est représenté par la figure V.13 où, les transitions x_i pour $i \in \{3, 4, 8\}$ sont les transitions d'entrée au processus du calcul d'itinéraire (c.-à-d. recevoir une requête en faisant appel aux services de la plateforme TransportML), et les transitions x_j avec $j \in \{6, 7, 9\}$ sont les transitions de sortie du processus du calcul d'itinéraire (c.-à-d. l'itinéraire est fourni par la plateforme). Le tableau V.3 décrit les différents éléments de la figure V.13.

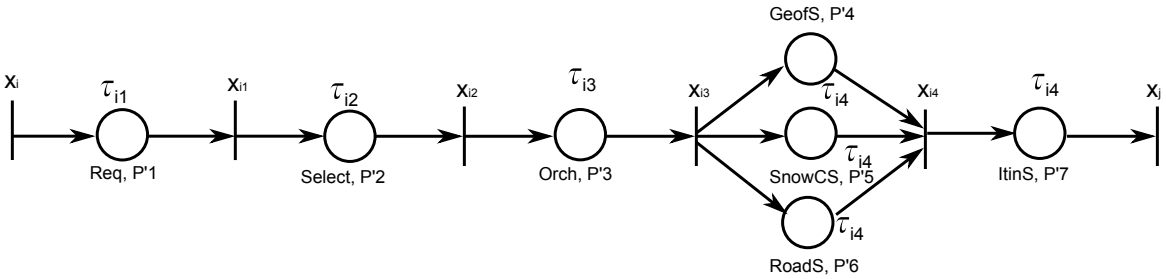


Figure V.13 – Représentation des places P_5, P_6 et P_{11} du modèle RdP de la figure V.12

Tableau V.3 – Légende du modèle RdP de la figure V.13

Transition/Place	Signification
x_i	Recevoir une requête de calcul d'itinéraire
P'_1	*RequestHandler* remplit les sections (*waypoints*, *vehicle* et *criteria*) du TMLDocument
x_{i1}	Fournir le *TMLDocument* rempli
P'_2	Sélectionner les services en fonction du QoS et la requête demandée
x_{i2}	Envoyer la liste des services sélectionnes
P'_3	Invoquer les services
x_{i3}	Envoyer le *TMLDocument*
P'_4	Traitement de la requête par le service *Geofencing*
P'_5	Traitement de la requête par le service *SnowClearance*
P'_6	Traitement de la requête par le service *RoadStatus*
x_{i4}	Rassembler les TMLDocuments en un seul et l'envoyer au service itinéraire
P'_7	Traitement de la requête par le service itinéraire
x_j	Envoyer itinéraire

Afin de décrire le comportement du scénario étudié par des équations (max,+), nous associons à chaque transition x_i le dateur $x_i(k) \in \mathbb{R}_{max}$ qui représente la date du $k^{ème}$ franchissement de la transition x_i. De même, nous associons à la transition d'entrée U (resp. de sortie Y) le dateur $U_1(k)$ (resp. $y_1(k)$). Les équations (max,+) décrivant le comportement du premier pattern (choix différé) du scénario de la figure V.12, modélisé par les transitions $<U, x_1, x_2>$ et la place $<P_1>$, sont ainsi données par :

$\forall k \geq 1$:

$$\begin{cases} x_1(k) & = f_{x_1}(k) \otimes \tau_1 \otimes u_1(k) \\ x_2(k) & = f_{x_2}(k) \otimes \tau_1 \otimes u_1(k) \end{cases} \tag{V.1}$$

Les expressions de $f_{x_i}(k)$ sont bien détaillées dans le chapitre 3, §3.3. Nous allons associer f_{x_1} à toutes les transitions issues de x_1 et f_{x_2} à toutes les transitions issues de x_2, ceci afin de respecter le franchissement réel et virtuel des transitions d'une branche donnée. L'équation (max,+) du pattern choix différé est ainsi généralisée par (V.2).

$$X_{choix}(k) = F_1(k) \otimes B_1 \otimes U(k) \tag{V.2}$$

Sachant que $X_{choix} \in \mathbb{R}_{max}^{2\times 1}$ est composé des deux transitions x_1 et x_2, $F_1(k)$ est la matrice de routage utilisée pour résoudre le conflit dans la place P_1 avec :

$F_1 \in \mathbb{R}_{max}^{2\times 2}$ et :

$$F_1(k) = \begin{pmatrix} f_{x_1}(k) & \epsilon \\ \epsilon & f_{x_2}(k) \end{pmatrix}$$

$B_1 \in \mathbb{R}_{max}^{2\times 1}$ avec $B_1^t = \begin{pmatrix} \tau_1 & \tau_1 \end{pmatrix}$. Finalement, $U \in \mathbb{R}_{max}^{1\times 1}$ et contient un seul élément associé à l'unique transition d'entrée.

Nous allons procéder de la même manière pour fournir les équations (max,+) des autres patterns. Pour le pattern multi-fusion représenté par les transitions $<x_1, x_2, x_3>$ et la place $<P_2>$, nous obtenons l'équation suivante :

$\forall k \geq 1$:

$$x_3(k) = f_{x_1}(k) \otimes \tau_2 \otimes x_1(k - m(Q_2)) \oplus f_{x_2}(k) \otimes \tau_2 \otimes x_2(k - m(Q_1)) \tag{V.3}$$

De manière générale (V.3) s'écrit sous la forme matricielle suivante :

$\forall k \geq k_i$, pour tout $i \in \{1, 2, 3\}$:

$$X_{fusion}(k) = F_2(k) \otimes A_1 \otimes X_{fusion}(k - k_i) \tag{V.4}$$

Sachant que $X_{fusion} \in \mathbb{R}_{max}^{3\times 1}$ est composé des transitions x_1, x_2 et x_3. $F_2 \in \mathbb{R}_{max}^{3\times 3}$ avec :

$$F_2(k) = \begin{pmatrix} \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & f_{x_1}(k) \oplus f_{x_2}(k) \end{pmatrix}$$

$F_2(k)$ est la matrice de routage utilisée pour déterminer pour chaque événement k, la transition franchie en fonction des franchissements réels ou virtuels des transitions x_1 et x_2.

$A_1 \in \mathbb{R}_{max}^{3\times3}$ avec :

$$A_1 = \begin{pmatrix} \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon \\ \tau_2 & \tau_2 & \epsilon \end{pmatrix}$$

$k_i = m(Q_i)$ pour $i = 1, 2, 3$ avec $k_3 = 0$, $k_1 = m(Q_2)$ et $k_2 = m(Q_1)$.

Les deux équations (V.2) et (V.4) sont réécrites et généralisées en une seule équation comme décrit dans ce qui suit.

$\forall k \geq k_i$:

$$X_{cf}(k) = F_2(k) \otimes A_1 \otimes X_{cf}(k - k_i) \oplus F_1(k) \otimes B_1 \otimes U(k) \tag{V.5}$$

A partir des vecteurs X_{choix} et X_{fusion} nous obtenons un seul vecteur $X_{cf} = (X_{choix}, X_{fusion}$ soit, $X_{cf}^t = \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix}$. Avec la nouvelle fusion, la matrice de routage F_1 a une taille égale à 3×3 et est définie par :

$$F_1(k) = \begin{pmatrix} f_{x_1}(k) & \epsilon & \epsilon \\ \epsilon & f_{x_2}(k) & \epsilon \\ \epsilon & \epsilon & \epsilon \end{pmatrix}$$

De même le vecteur B_1 de l'équation (V.2) devient :

$B_1 \in \mathbb{R}_{max}^{3\times1}$ avec $B_1^t = \begin{pmatrix} \tau_1 & \tau_1 & \epsilon \end{pmatrix}$.

Dans la nouvelle équation nous gardons la même notation B_1. Les autres éléments restent inchangés.

Les deux patterns fractionnement parallèle, représenté par les transitions $<x_2, x_3, x_4, x_5>$ et les places $<P_2, P_3, P_4>$, et synchroniseur, représenté par les transitions $<x_4, x_5, x_8>$ et les places $<P_7, P_8>$, sont décrits par les équations (V.6).

$\forall k \geq 1$:

$$\begin{cases} x_3(k) &= f_{x_2}(k) \otimes \tau_2 \otimes x_2(k) \\ x_4(k) &= f_{x_2}(k) \otimes \tau_3 \otimes x_2(k) \\ x_5(k) &= f_{x_2}(k) \otimes \tau_4 \otimes x_2(k) \\ x_7(k) &= f_{x_2}(k) \otimes \tau_6 \otimes x_4(k) \\ x_8(k) &= f_{x_2}(k) \otimes [\tau_7 \otimes x_4(k) \oplus \tau_8 \otimes x_5(k)] \end{cases} \tag{V.6}$$

Il est important de noter que la transition x_3 est une transition commune pour les deux patterns multi-fusion et fractionnement parallèle. Le dateur $x_3(k)$, pour chaque k, a donc une expression propre (l'équation (V.3)) au pattern fusion, et une deuxième expression (la première

équation du système (V.6)) propre au pattern fractionnement parallèle. Compte tenu du conflit structurel de la place P_1, une des deux transitions x_1 ou x_2 est franchie à chaque fois qu'un jeton arrive dans la place P_1. Ceci signifie que pour chaque k ($k^{ème}$ jeton arrivé dans P_1) une des expressions (V.3) où la première équation de (V.6) s'annule. L'expression (max,+) de la combinaison des deux patterns est donnée donc par la somme des deux expressions comme nous le verrons plus tard dans l'équation (V.9). Nous obtenons le modèle matriciel général.

$$X_{fs}(k) = F_3(k) \otimes A_2 \otimes X_{fs}(k) \tag{V.7}$$

Où les expressions des X_{fs}, F_3 et A_2 sont données dans l'équation (V.8).

$$\begin{pmatrix} x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_7 \\ x_8 \end{pmatrix}(k) = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & f_{x_2}(k) & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & f_{x_2}(k) & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & f_{x_2}(k) & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2}(k) & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2}(k) \end{pmatrix} \otimes \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \tau_2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \tau_3 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \tau_4 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \tau_6 & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \tau_7 & \tau_8 & \epsilon & \epsilon \end{pmatrix} \otimes \begin{pmatrix} x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_7 \\ x_8 \end{pmatrix}(k) \tag{V.8}$$

De la même manière, les équations (V.7) et (V.5) seront regroupées en une seule équation (V.9).

On a, $\forall k \geq 1$:

$$X_{cfr}(k) = F_3(k) \otimes A_2 \otimes X_{cfr}(k) \oplus F_2(k) \otimes A_1 \otimes X_{cfr}(k - k_i) \oplus F_1(k) \otimes B_1 \otimes U(k) \tag{V.9}$$

Avec :

– $X^t_{cfr} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_7 & x_8 \end{pmatrix}$

– On garde le même nom de la matrice F_3 mais sa taille augmente et son expression devient,

$$F_3 = \begin{pmatrix} f_{x_1} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} \end{pmatrix}$$

– La taille de la matrice A_2 augmente et son expression devient :

$$A_2 = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \tau_2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \tau_3 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \tau_4 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \tau_6 & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \tau_7 & \tau_8 & \epsilon & \epsilon \end{pmatrix}$$

– De même pour la matrice F_2, sa taille augmente et son expression devient :

$$F_2 = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_1} \oplus f_{x_2} \end{pmatrix}$$

– De la même manière, la taille de la matrice A_1 augmente et son expression devient :

$$A_1 = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \tau_2 & \tau_2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix}$$

– On a $F_1(k) = F_3(k)$

– $B_1^t = \begin{pmatrix} \tau_1 & \tau_1 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix}$

Nous allons procéder de la même manière pour le reste des patterns à savoir, le pattern sequence représenté par les transitions $<x_3, x_6>$ et la place $<P_5>$, le pattern synchroniseur, représenté par les transitions $<x_6, x_7, x_9, x_{10}>$ et les places $<P_9, P_{10}, P_{12}>$, et le pattern choix différé, constitué des transitions $<x_6, x_{10}, x_{11}>$ et la place $<P_9>$.

$\forall k \geq 1$:

$$\begin{cases} x_6(k) & = f_{x_1}(k) \otimes \tau_5 \otimes x_3(k) \\ x_{10}(k) & = f_{x_{10}} \otimes (k)\tau_9 \otimes x_6(k) \\ x_{11}(k) & = f_{x_{11}} \otimes (k)\tau_9 \otimes x_6(k) \end{cases} \tag{V.10}$$

$\forall k \geq 1$:

$$x_{10}(k) = f_{x_{10}}(k) \otimes \tau_9 \otimes x_6(k) \oplus f_{x_2}(k) \otimes \tau_{10} \otimes x_7(k) \oplus f_{x_2}(k) \otimes \tau_{12} \otimes x_9(k) \tag{V.11}$$

Enfin, la sortie du modèle est exprimée par l'équation suivante :

$\forall k \geq m(Q_i)$, avec $i = 3, 4$:

$$y(k) = \tau_{13} \otimes x_{10}(k - m(Q_3)) \oplus \tau_{13} \otimes x_{11}(k - m(Q_4)) \tag{V.12}$$

Sachant que $f_{x_{10}}(k) = f_{x_2}(k)$ et $f_{x_{11}}(k) = f_{x_1}(k)$ et $k \geq m(Qi)$ pour $i = 3, 4$, nous obtenons le système d'équations du modèle général de la figure V.12 en fusionnant toutes les équations de chaque pattern, nous obtenons l'équation (V.13).

Pour tout $k \geq k_i$:

$$\begin{cases} X(k) & = F(k) \otimes A \otimes X(k) \oplus F(k) \otimes A_1 \otimes X(k - k_i) \oplus F(k) \otimes B \otimes U(k) \\ Y(k) & = C_{k_l} \otimes X(k - k_l) \end{cases} \tag{V.13}$$

La fusion des expressions du dateur $x_{10}(k)$ données par la seconde équation de (V.10) et par (V.11) se fait de la même manière que le dateur $x_3(k)$ comme indiqué précédemment.

Remarque 4.2.1 *La résolution et l'arbitrage des conflits structurels du modèle RdP, nous a conduit à un système d'équations (max,+)-linéaire non stationnaire. Les matrices caractéristiques du modèle (V.13) sont à coefficients variables. Cet aspect non-stationnaire est dû aussi aux temporisations variables associées aux places du modèle comme nous le verrons plus tard dans la section 5.1.1.*

Les différents éléments du système (V.13) sont donnés par :

- Le vecteur $X(k) \in \mathbb{R}_{max}^{11\times 1}$ (resp. $U(k) \in \mathbb{R}_{max}^{1\times 1}$ et $Y(k) \in \mathbb{R}_{max}^{1\times 1}$), est composé des transitions internes du modèle RdP et représente le vecteur d'état (resp. le vecteur d'entrée et le vecteur de sortie du système).
- La matrice de routage $F(k) \in \mathbb{R}_{max}^{11\times 11}$ est définie par :

$$F = \begin{pmatrix} f_{x_1} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & f_{x_1} \oplus f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_1} \oplus f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_2} & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & f_{x_1} \end{pmatrix}$$

- $B \in \mathbb{R}_{max}^{11\times 1}$, où $B^t = \begin{pmatrix} \tau_1 & \tau_1 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix}$
- $A_1 \in \mathbb{R}_{max}^{11\times 11}$ avec $A_1 = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \tau_2 & \tau_2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix}$
- $C_{k_l} \in \mathbb{R}_{max}^{1\times 11}$ avec l = 10, 11 :
  $C_{k_{10}} = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \tau_{13} & \epsilon \end{pmatrix}$ et $C_{k_{11}} = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \tau_{13} \end{pmatrix}$
- $\forall i \in \{3, ..., 11\}$, $k_i = 0$ sauf pour $k_1 = m(Q_2)$ et $k_2 = m(Q_1)$
- $\forall l \in \{1, ..., 9\}$, $k_l = 0$ sauf pour $k_{10} = m(Q_4)$ et $k_{11} = m(Q_3)$

– $A \in \mathbb{R}_{max}^{11\times11}$ avec $A = \begin{pmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \tau_2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \tau_3 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \tau_4 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \tau_5 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \tau_6 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \tau_6 & \tau_8 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \tau_{11} & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \tau_9 & \tau_{10} & \epsilon & \tau_{12} & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \tau_9 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix}$

La première équation du système (V.13) s'écrit sous la forme suivante :

$$X(k) = F(k) \otimes A \otimes X(k) \oplus F(k) \otimes B' \otimes U(k) \tag{V.14}$$

En effet, dans l'exemple choisi, le terme en $F(k) \otimes A_{k_i} \otimes X(k-k_i)$ de l'équation (V.13), se voit simplifié puisque pour chaque franchissement de U_1, x_3 sera franchie réellement suite au franchissement de x_1 ou x_2. Nous remplaçons donc dans l'expression de $x_3(k)$, $x_1(k)$ et $x_2(k)$ par $\tau_1 \otimes U(k)$. B' devient :

$$B'^t = \begin{pmatrix} \tau_1 & \tau_1 & \tau_1 \otimes \tau_2 & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix}$$

L'équation (V.14) devient :

$$X(k) = A(k) \otimes X(k) \oplus B'(k) \otimes U(k) \tag{V.15}$$

Avec $A(k) = F(k) \otimes A$, $B'(k) = F(k) \otimes B'$ où :
$B'(k)^t = \begin{pmatrix} \tau_1 \otimes f_{x_1}(k) & \tau_1 \otimes f_{x_2}(k) & \tau_1 \otimes \tau_2 \otimes [f_{x_1}(k) \oplus f_{x_2}(k)] & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{pmatrix}$. Notons que $\forall k \geq 1$ on a $f_{x_1}(k) \oplus f_{x_2}(k) = e$.

Dans le but d'évaluer les dates de réponses des différents services pour chaque requête, nous procédons à la résolution du système (V.15). Il s'agit d'une équation implicite, sa résolution nécessite le calcul de l'étoile de Kleene A^* dont l'expression est donnée par $A(k)^* = I \oplus A^1 \oplus A^2 \oplus ... \oplus A^n \oplus ...$ où I représente la matrice identité de même dimension que la matrice A et dont la diagonale est constituée de l'élément neutre e et ϵ partout ailleurs.

$A(k)^* \in \mathbb{R}_{max}^{11\times11}$, dans notre cas, on a $A(k)^* = I \oplus A(k)^1 \oplus A(k)^2 \oplus A(k)^3 \oplus A(k)^4$. L'expression de $A(k)^*$ est donnée comme suit, sachant que les coefficients f_{x_i} dépendent également de k (f_{x_i}

$= f_{x_i}(k))$:

$$
A(k)^* = \begin{pmatrix}
e & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\
\epsilon & e & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\
\epsilon & \tau_2 & e & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\
\epsilon & \tau_3 \otimes f_{x_2} & \epsilon & e & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\
\epsilon & \tau_4 \otimes f_{x_2} & \epsilon & \epsilon & e & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\
\epsilon & \tau_5 \otimes \tau_2 & \tau_5 & \epsilon & \epsilon & e & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\
\epsilon & \tau_6 \otimes \tau_5 \otimes f_{x_2} & \epsilon & \tau_6 \otimes f_{x_2} & \epsilon & \epsilon & e & \epsilon & \epsilon & \epsilon & \epsilon \\
\epsilon & f_{x_2} \otimes [\tau_6 \otimes \tau_3 \oplus \tau_8 \otimes \tau_4] & \epsilon & \tau_6 \otimes f_{x_2} & \tau_8 \otimes f_{x_2} & \epsilon & \epsilon & e & \epsilon & \epsilon & \epsilon \\
\epsilon & [\tau_8 \otimes \tau_4 \oplus \tau_6 \otimes \tau_3] \otimes \tau_{11} \otimes f_{x_2} & \epsilon & \tau_{11} \otimes \tau_6 \otimes f_{x_2} & \tau_{11} \otimes \tau_8 \otimes f_{x_2} & \epsilon & \epsilon & \tau_{11} \otimes f_{x_2} & e & \epsilon & \epsilon \\
\epsilon & [\tau_9 \otimes \tau_5 \otimes \tau_2 \oplus \tau_{10} \otimes \tau_6 \otimes \tau_3] \otimes f_{x_2} \oplus [\tau_8 \otimes \tau_4 \oplus \tau_6 \otimes \tau_3] \otimes \tau_{12} \otimes \tau_{11} \otimes f_{x_2} & \tau_9 \otimes \tau_5 \otimes f_{x_2} & \tau_6 \otimes f_{x_2} \otimes [\tau_{11} \otimes \tau_{12} \oplus \tau_{10}] & \tau_{12} \otimes \tau_{11} \otimes \tau_8 \otimes f_{x_2} & \tau_9 \otimes f_{x_2} & \tau_{10} \otimes f_{x_2} & \tau_{12} \otimes f_{x_2} \otimes \tau_{11} & \tau_{12} \otimes f_{x_2} & e & \epsilon \\
\epsilon & \tau_5 \otimes f_{x_1} \otimes \tau_2 \otimes \tau_9 & \tau_9 \otimes \tau_5 \otimes f_{x_1} & \epsilon & \epsilon & \tau_9 \otimes f_{x_1} & \epsilon & \epsilon & \epsilon & \epsilon & e
\end{pmatrix}
$$

En utilisant cette matrice, la plus petite solution explicite de l'équation (V.15) est donnée par l'équation (V.16).

$\forall k \geq 1$:

$$X(k) = A(k)^* \otimes B'(k) \otimes U(k) \tag{V.16}$$

La matrice $A(k)^* \otimes B'(k)$ est donnée par (V.17).

$$A(k)^* \otimes B'(k) = \begin{pmatrix} \tau_1 f_{x_1} \\ \tau_1 f_{x_2} \\ \tau_1 \tau_2 \\ \tau_1 \tau_3 f_{x_2} \\ \tau_1 \tau_4 f_{x_2} \\ \tau_1 \tau_2 \tau_5 \\ \tau_1 \tau_5 \tau_6 f_{x_2} \\ (\tau_3 \tau_6 \oplus \tau_4 \tau_8) \tau_1 f_{x_2} \\ (\tau_3 \tau_6 \oplus \tau_4 \tau_8) \tau_1 \tau_{11} f_{x_2} \\ f_{x_2} \tau_1 (\tau_2 \tau_5 \tau_9 \oplus \tau_{11} \tau_{12} (\tau_3 \tau_6 \oplus \tau_4 \tau_8) \oplus \tau_3 \tau_6 \tau_{10}) \\ \tau_1 \tau_2 \tau_5 \tau_9 f_{x_1} \end{pmatrix} \tag{V.17}$$

Les coefficients f_{x_i} dépendent de k aussi : $f_{x_i} = f_{x_i}(k)$.

Par conséquent, la sortie du système est donnée par :

$$Y(k) = C_{k_{10}} A(k)^* B'(k) U(k - k_{10}) \oplus C_{k_{11}} A(k)^* B'(k) U(k - k_{11}) \tag{V.18}$$

Avec :

- $C_{k_{10}} A(k)^* B'(k) = f_{x_2}(k) \tau_1 \tau_{13} (\tau_2 \tau_5 \tau_9 \oplus \tau_{11} \tau_{12} (\tau_3 \tau_6 \oplus \tau_4 \tau_8) \oplus \tau_3 \tau_6 \tau_{10})$
- $C_{k_{11}} A(k)^* B'(k) = \tau_1 \tau_2 \tau_5 \tau_9 \tau_{13} f_{x_1}(k)$
- $k_{10} = m(Q_3)$ et $k_{11} = m(Q_4)$. Sachant que dans le cas des franchissements virtuels des transitions, l'ajout d'un jeton virtuel dans la place Q_3 (resp. Q_4) ne changera pas la valeur de k_{10} (resp. k_{11}).

Les deux équations (V.16) et (V.18) nous permettent d'évaluer les dates de réponse de différents services invoqués pour répondre aux différentes requêtes reçues par le *service permanence*. Ces dates dépendent des dates d'arrivée des requêtes et des temps de réponse de services. Nous rappelons également que l'information sur k nous renseigne la $k^{ème}$ requête reçue. Nous allons étudier de plus prêt le temps de réponse en fonction de la $k^{ème}$ requête reçue de façon séquentielle par le système. La section 5.1 est consacrée à cette étude et les résultats obtenus seront comparés aux résultats issus de l'exécution du système développé.

5 Expérimentations et résultats

Nous présentons dans ce chapitre les différentes applications développées dans le cadre du projet ASSET. Nous signalons que ces applications et leurs services Web associés sont des créations ex-nihilo pour montrer la faisabilité et l'importance d'une plateforme de collaboration de services. Pour ce faire, nous illustrons à travers l'étude de cas décrite dans la section précédente les points forts de la plateforme à partir de son analyse et de son évaluation formelles. L'objectif étant non seulement d'aider à la compréhension de notre approche de composition de services mais aussi, de démontrer sa faisabilité et son efficacité à partir d'une modélisation formelle basée sur les workflow patterns et l'algèbre (max,+). Une étude complète a été réalisée à partir de la spécification du scénario en passant par sa modélisation formelle jusqu'à son implémentation et sa réalisation, et enfin sa validation à partir des expérimentations de différentes applications. Nous allons scinder les expérimentations en deux parties. La première partie consiste à étudier et analyser le modèle mathématique du prototype en réalisant des simulations sous Matlab et de comparer les résultats obtenus à ceux qui relèvent d'une simulation effectuée en exécutant les applications développées. La deuxième partie des expérimentations se focalise sur les tests réalisés sur le terrain en temps réels, dont les résultats seront élucidés et commentés.

5.1 Expérimentation par simulation

5.1.1 Simulation des modèles (max,+)

Les modèles (max,+) obtenus dans la section 4.2 permettent, pour un ensemble de paramètres donnés du scénario, de déterminer le temps de réponse nécessaire pour répondre à une requête d'un utilisateur. Il est important de noter que, dans la pratique, le temps de réponse de chaque service Web dépend de plusieurs facteurs dont la disponibilité et la saturation du réseau. Compte tenu de ces facteurs, le temps de réponse varie d'une requête à une autre. Afin d'intégrer cette information dans le modèle (max,+), nous supposons que le temps de réponse de chaque service sélectionné selon sa qualité pour répondre à une requête est compris dans un intervalle de temps avec une borne inférieure (réponse dans les meilleurs des cas), et une borne supérieure (réponse dans le pire des cas). Souvent, pour un RdP P-temporisé (ou un graphe d'événements temporisés GET), on choisit la valeur moyenne de l'intervalle.

Les données du système sont représentées dans le tableau V.4. Nous avons choisi des valeurs variables pour les temporisations associées aux places. En effet, pour tout $\tau_i \in [\alpha_i, \beta_i]$ avec $\alpha_i < \beta_i$ et $i \in \{1, ..., 13\}$. Sachant que α_i (resp. β_i) correspond au temps minimal (resp. maximal) nécessaire à l'exécution de la tâche représentée par la place P_i. La variance de τ_i est

Tableau V.4 – Les intervalles temporels associés aux places du scénario en ms

τ_i	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_{10}	τ_{11}	τ_{12}	τ_{13}
α_i	0	0	0	1	1101	1111	31	1	3	1	1135	1	34
β_i	3	10	10	20	1448	1412	56	9	10	12	1360	50	64

justifiée par le fait que ces valeurs dépendent de l'environnement réel dans lequel le scénario est réalisé. Cependant, pour être proche du cas réel, les valeurs maximales et minimales sont fournies à partir de l'exécution du scénario à plusieurs reprises. Les données du tableau V.4 sont intégrées dans le modèle mathématique (V.18) qui, est ensuite simulé sous Matlab. Nous soulignons que le franchissement de la transition U suit une loi uniforme et pour le calcul du temps de réponse pour chaque requête se base sur la connaissance de l'entrée du système. La loi uniforme considérée dans ces expérimentations est choisie juste à titre d'exemple pour illustrer l'approche proposée. Une autre loi, plus proche de la réalité, peut être choisie ainsi les modèles et le raisonnement restent les mêmes. La sortie du système dépend, entre autres, de son entrée.

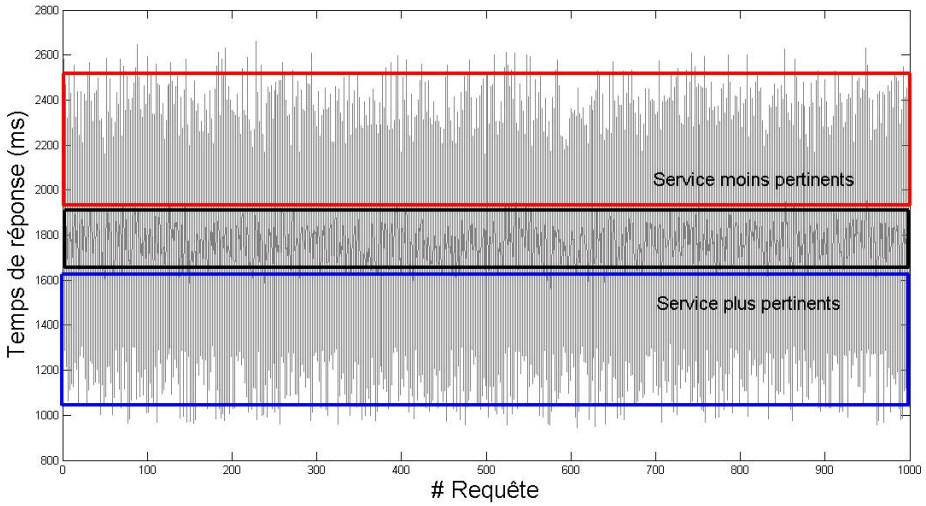


Figure V.14 – Temps de réponse avec les valeurs aléatoires des temporisations

La figure V.14 représente les résultats de simulation du modèle (max,+) du scénario où, les valeurs des temporisations sont calculées de façon aléatoire dans leurs intervalles respectifs. Cette figure illustre les temps de réponse nécessaires pour traiter un ensemble de requêtes. On remarque que l'allure du graphe est variable entre des valeurs minimales et maximales. Le temps de réponse à une requête s'approche de sa valeur maximale quand par exemple un service met plus de temps pour répondre à la tâche qui lui a été assignée ou une autre raison, qui relève de la surcharge du réseau comme indiqué précédemment. Nous constatons que le graphe est dense

au tour de la valeur $\simeq$ 1800 ms du temps de réponse, ce qui représente le temps de réponse moyen des requêtes obtenues à partir des valeurs moyennes des temporisations τ_i.

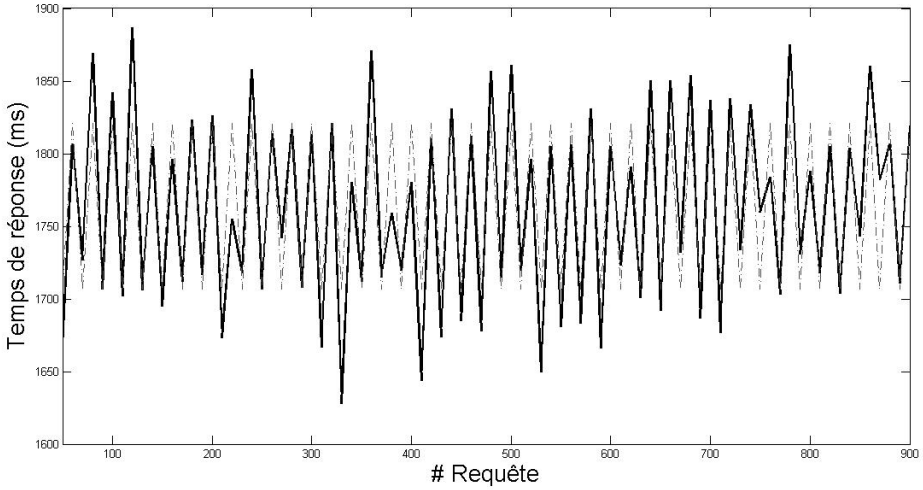


Figure V.15 – Temps de réponse avec les valeurs moyennes et aléatoires des temporisations

La figure V.15 représente les graphes des temporisations moyennes et aléatoires associées aux services du scénario. Par simulation on observe un comportement monotone du temps de réponse moyen en fonction de l'arrivée des requêtes. En effet les valeurs change de façon périodique entre une valeur maximale et une valeur minimale. Ces deux valeurs correspondent, mis à part aux conditions de l'environnement du scénario, à l'appel des trois services *Police*, *Ambulance* et *Hôpital* en simultané ou juste à l'appel unique du service *Police*. Nous remarquons que lorsque le service *permanence* envoie la requête aux trois services, le temps de réponse est plus grand (au dessus des 1800 ms), ce comportement se justifie par le fait que le service *Ambulance* va demander deux fois le calcul de l'itinéraire, ce qui demande un plus du temps pour fournir la réponse finale à l'utilisateur. Notons que le comportement de l'évolution du scénario obtenu à partir du modèle formel, est pratiquement identique à celui obtenu par des simulations réels sur machine comme illustré dans la section suivante (figure V.16).

5.1.2 Simulations par exécution sur machine

Les simulations effectuées sur machine ne tiennent pas compte de la latence du réseau. Les requêtes sont formulées en modifiant à chaque fois les entrées (c.-à-d. le lieu de l'incident, et le type de l'incident). La figure V.16 montre le temps de réponse en fonction de la $k^{ème}$ requête.

Nous remarquons que le graphe est plus dense lorsque le temps de réponse est proche de la moyenne qui est approximativement $\simeq 1771ms$. Des valeurs sont dispersées et se rapprochent

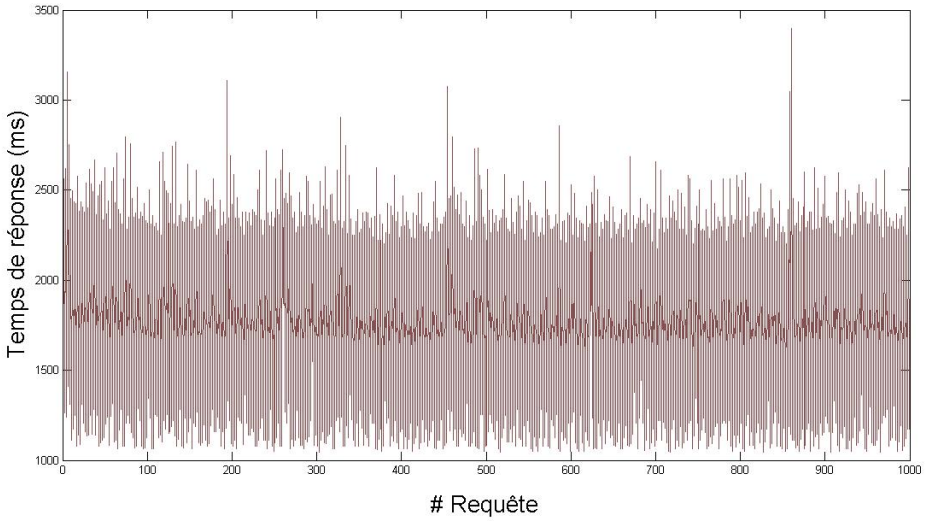


Figure V.16 – Temps de réponse obtenu par simulation sur machine

des valeurs maximales, elles correspondent au temps de réponse des services composés qui, mettent plus de temps à répondre à une requête. Ces temps de réponse varient entre 1148 ms et 2395 ms lorsque, dans le premier cas seul le service *Police* est appelé et dans le deuxième cas les trois services (*police*, *ambulance* et *hôpital*) sont invoqués en parallèle.

En comparant les résultats obtenus par les deux méthodes considérées à savoir la solution du modèle (max,+) et la simulation du scénario sur machine, nous constatons que ces résultats sont les mêmes. En effet, les deux figures V.14 et V.16, correspondent respectivement aux deux méthodes, fournissent les mêmes informations. Le temps de réponse moyen pour les deux courbes est d'environ 1800 ms. Ceci montre une adéquation parfaite entre l'exécution du scénario dans un contexte réel et son modèle mathématique.

5.2 Expérimentations sur le terrain

Nous allons présenter dans cette partie les tests réels réalisés sur le terrain. Les résultats illustrent les avantages de l'utilisation d'une plateforme de collaboration de services. En effet, dans ces expérimentations nous nous sommes intéresses à l'étude des performances de la plateforme en étudiant le scénario du calcul d'itinéraire. Ce prototype a été développé et testé comme preuve de concept. Il est également testé dans les zones urbaines et sur autoroutes afin de montrer l'avantage d'utiliser cette plateforme pour le partage d'informations liées à la route ainsi que pour l'interaction entre services.

Le service itinéraire calcule l'itinéraire le plus court entre deux lieux géographiques distants

en utilisant les informations fournies par les services décrits précédemment. Ce processus se déroule en deux étapes. Pour commencer, les caractéristiques du véhicule et les points intermédiaires (c.-à-d. *waypoints*) sont définis et envoyés à la plateforme. La requête est ensuite traitée en invoquant les différents services nécessaires pour définir l'itinéraire approprié. Autrement dit, chaque fois que la plateforme TransportML reçoit la demande d'un client (p. ex. "Je suis à la recherche d'un itinéraire pour partir d'un point A vers un point B"). Le service orchestration va appeler ces services afin d'obtenir le document *TMLDocument* avec tous les éléments requis. Une fois que chaque service invoqué traite la tâche qui lui a été assignée, le document TML où les paramètres ***<unadvisable_area>*** et ***<advised_areas>*** sont renseignés, est fourni au service orchestration. Ce dernier va rassembler tous les documents TML, reçus des différents services invoqués, en un seul document. L'étape suivante consiste à l'invocation du service itinéraire pour calculer l'itinéraire demandé en se basant sur le document TML résultant de l'invocation en amont des autres services. Le résultat final est ensuite envoyé à l'utilisateur.

Les services invoqués pour le calcul d'itinéraire sont le service déneigement qui fournit les routes non déneigées en temps réel, le service état des routes permettant de fournir les portions de routes où il y a les travaux et finalement, le service geofencing permettant de fournir des zones interdites pour un type de véhicule.

5.2.1 Description du cadre des tests

Les différents scénarios développés précédemment ont fait objet de tests réels sur le terrain. Pour ce faire, nous avons utilisé une voiture, et pour certains scénarios deux voitures, équipées d'un ordinateur, d'un dispositif GPS et d'un modem 3G. Ces tests ont eu lieu dans une zone urbaine et sur l'autoroute du territoire de Belfort. Nous avons mis en place un serveur accessible depuis internet contenant toutes les applications décrites au début de ce chapitre. Chaque échange avec les véhicules se fait via ce serveur.

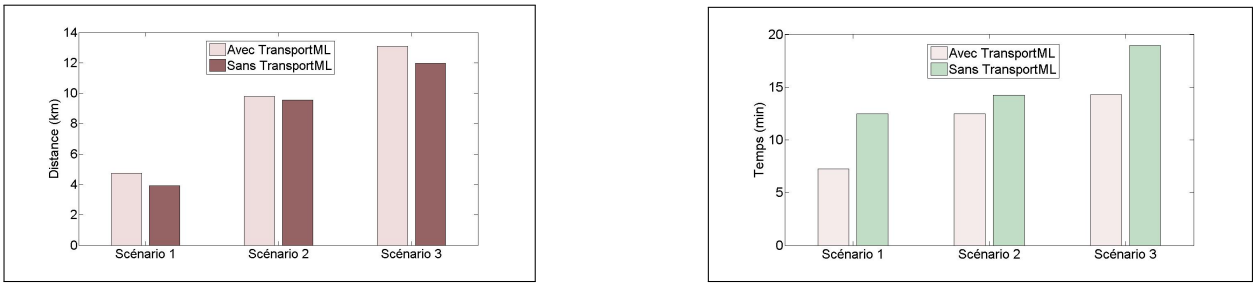
Nous avons, d'une part, analysé la latence relative à l'utilisation de la technologie 3G, nous avons également étudié les performances des différentes applications sur le serveur. Les résultats liés à ces paramètres sont décrits dans la section suivante. Pendant les expérimentations, la vitesse des véhicules et leurs positions GPS sont recueillies toutes les 3 secondes. Le temps de traitement au niveau de chaque service et chaque application est aussi enregistré.

5.2.2 Résultats

Distance et temps parcourus avec et sans TransportML

Pour mettre en évidence l'impact de l'utilisation de TransportML, nous avons comparé la

distance et le temps nécessaire pour aller du point de départ au point de destination, dans les deux cas avec et sans l'utilisation de TransportML. Trois scénarios sont considérés dans les expériences. Dans le premier scénario nous invoquons un seul service, dans le deuxième scénario nous appelons deux services, et enfin dans le troisième scénario trois services sont invoqués. Pour chaque scénario, les points de départ et d'arrivée sont différents. Les résultats illustrés dans la figure V.17a montrent que, dans tous les scénarios, lorsque nous utilisons TransportML, la distance parcourue est supérieure à celle parcourue lorsque TransportML n'est pas utilisé. Cependant, la figure V.17b montre que le temps requis pour le même chemin de déplacement est plus faible lorsque TransportML est utilisé.



(a) Distance de l'itinéraire **(b)** Durée de l'itinéraire

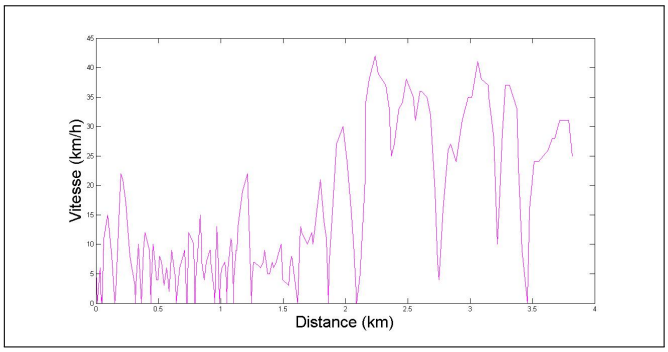
Figure V.17 – Distance parcourue et temps associé avec et sans TransportML

En fait, en utilisant TransportML, les routes qui ne sont pas encore déneigées et celles qui sont en travaux sont ainsi évitées. Par conséquent, la distance parcourue pour atteindre une destination donnée est plus grande que d'habitude (absence de travaux, de la neige et de zones interdites), en revanche la quantité de temps nécessaire pour atteindre la même destination et plus faible. Plus précisément, sans utiliser TransportML, le temps est plus long puisque les véhicules, traversant les zones déconseillées, subissent un ralentissement (embouteillage, vitesse limitée, feux rouge avec circulation alternée de véhicules sur des tronçons de routes à cause des travaux, neige ...). Les résultats obtenus montrent que le temps moyen de déplacement est d'environ 27% réduit lorsque TransportML est utilisé.

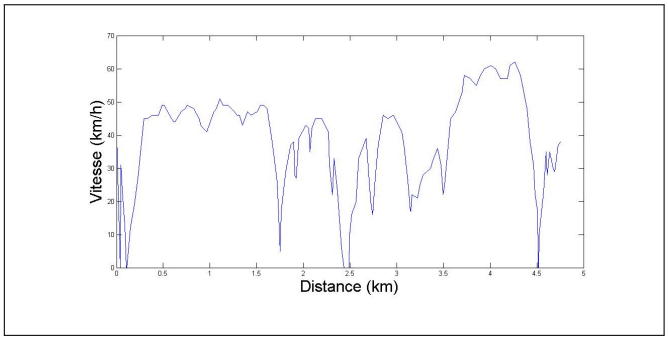
Vitesse en fonction de la distance avec et sans TranportML

Nous avons mesuré les habitudes de conduire en recueillant la vitesse du véhicule et la position depuis le dispositif GPS embarqué afin de calculer le temps et la distance parcourus.

La figure V.18 montre la variation de la vitesse du véhicule en fonction de la distance parcourue avec et sans l'utilisation de TranportML. La figure V.18a montre que, sans TransportML, les ralentissements, voir les arrêts, du véhicule sont remarquables. En d'autres termes,



(a) Sans TransportML



(b) Avec TransportML

Figure V.18 – Vitesse en fonction de la distance avec et sans utilisation de TransportML

dans la zone déconseillée, l'embouteillage commence à apparaître dès le début de la zone, et la vitesse du véhicule augmente/diminue lentement, ce qui conduit à une augmentation de l'accélération/décélération (environ sur 2 km). Ceci correspond à la congestion dont la zone est définie comme déconseillée. Toutefois, lorsque nous utilisons TransportML (figure V.18b), la vitesse du véhicule augmente, mais diminue certaine fois. Les véhicules se déplacent à une vitesse relativement constante (sauf en stop ou aux feux), et aucun comportement de congestion n'a été détecté. Lors de ces expériences, nous avons conclu que, la congestion du trafic causée principalement par la capacité des routes, des incidents de circulation, les zones de travaux, les conditions météorologiques, devraient être évités afin de réduire le temps de déplacement des véhicules d'urgence.

Temps d'appel de services

De même nous avons évalué le temps d'appel des services contre le temps de communication. En effet, ce temps de communication est le temps entre l'envoie de la requête et la reception de la réponse, sans tenir compte du temps de traitement de la requête. Les résultats sont illustrés par la figure V.19. Lorsque TransportML est utilisé, ce qui signifie que plusieurs services peuvent

être invoqués, le coût en temps nécessaire pour appeler ceux qui sont impliqués dans le scénario augmente légèrement malgré que ces services sont appelés séquentiellement. La durée moyenne de communication augmente aussi légèrement avec le nombre de services puisque la taille des documents TML augmente également. En fait, le temps nécessaire pour envoyer le document TML est proportionnel à sa taille et à la bande passante disponible.

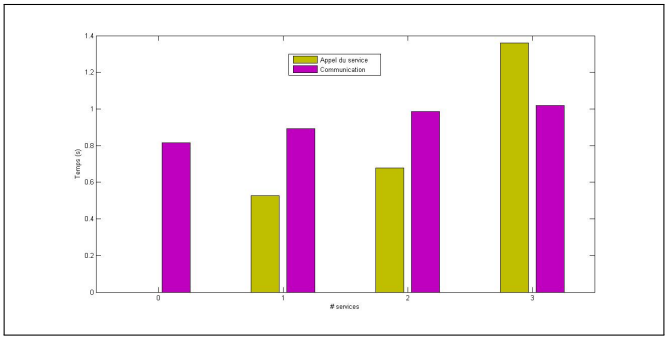
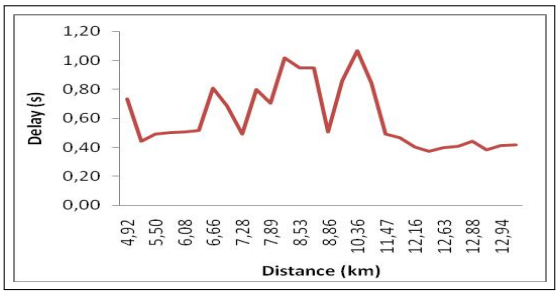


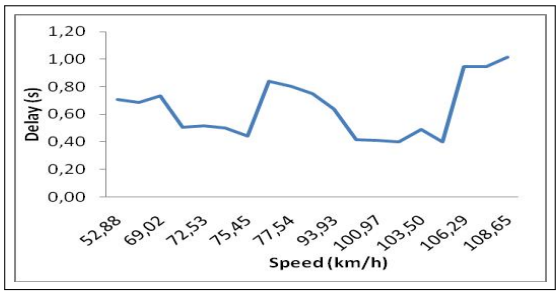
Figure V.19 – Temps de communication et d'appel de services

Délai de la communication V2I

Pour montrer la sensibilité de la communication 3G, nous avons mené des expériences sur l'autoroute sur un tronçon de 13km. Le délai pour envoyer des informations (par exemple, vitesse, position, direction) de l'ordinateur embarqué dans le véhicule à l'infrastructure (c.-à-d. serveur) a été mesuré. Les résultats obtenus montrent que la technologie 3G n'est ni sensible à la distance, à partir du véhicule à l'emplacement du serveur, comme indiqué dans la figure V.20a, ni à la vitesse du véhicule (figure V.20b). La durée moyenne de la communication (V2I ou I2V)est d'environ 0.6s.



(a) Distance V2I



(b) Vitesse du véhicule

Figure V.20 – Délai de la communication contre la distance et la vitesse

Le passage à l'échelle (*scalability*) de la plateforme

Nous avons également étudié l'évolutivité (en anglais scalability) de la plateforme. Cette mesure désigne la capacité de la plateforme à s'adapter au changement d'ordre de grandeur de la demande (croissant). En particulier sa capacité à maintenir ses fonctionnalités et ses performances en cas de forte demande.

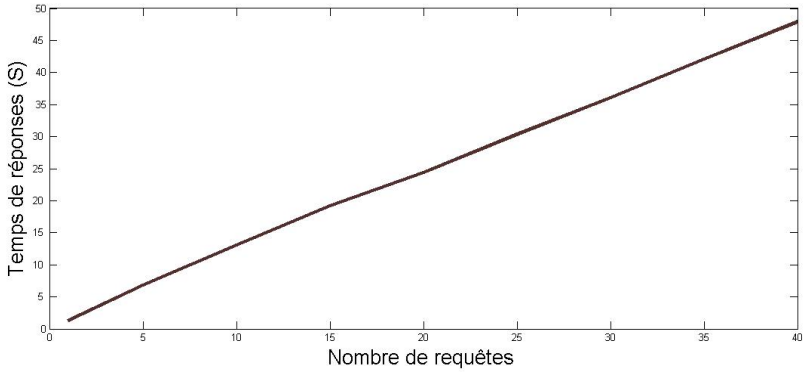


Figure V.21 – Le passage à l'échelle de la plateforme en fonction du nombre de requêtes

La figure V.21 illustre la mise à l'échelle de la plateforme en fonction du nombre de demandes en simultané. Nous pouvons déduire que le temps augmente de façon linéaire quand le nombre de requêtes augmente. Ceci représente une limite de la plateforme et constitue un des paramètres à prendre en considération lors des évolutions futures de la plateforme.

6 Conclusion

Dans ce chapitre nous avons présenté tous les scénarios développés dans le cadre du projet Européen ASSET. Nous avons également exposé le prototype réalisé et étudié afin de montrer la faisabilité et les bénéfices que pourraient apporter notre approche. Le scénario d'urgence a été définit et implémenté comme preuve de concept de la collaboration et l'interaction de manière automatique des LBS. Nous avons étudié de façon formelle et par expérimentation les performances de ce scénario. Dans un premier temps, nous avons modélisé le système à l'aide des RdP P-temporisés. Le modèle obtenu représente l'agrégation des patterns définis dans le chapitre III. Ensuite, nous avons représenté le modèle général sous forme d'équations (max,+) en se basant sur l'équation de chaque pattern. Les performances du modèle formel ont été analysées par le biais d'un ensemble de propriétés quantitatives et qualitatives vérifiées à l'aide des modèles (max,+) et RdP, la faisabilité de composition de services Web sont alors garantis. Ceci est vérifié à travers des simulations réalisés en utilisant la solution du modèle formel et

celles issues de l'exécution du scénario sur machine. Les résultats ont montrés que le modèle formel ainsi que le modèle expérimental se rapproche voire similaire.

Le prototype a été développé et des expériences sur le terrain ont été menées dans l'environnement urbain et sur l'autoroute. Les résultats sont présentés pour démontrer l'efficacité et l'impact de la collaboration interservices pour réduire le temps d'intervention des services d'urgence en cas d'incident. Les résultats obtenus ont montré que la durée moyenne de déplacement du véhicule d'urgence de son emplacement de départ vers l'endroit de l'incident est d'environ 27% réduit lorsque TransportML est utilisé, avec un léger surcoût. Cette diminution de la durée du voyage est due à l'itinéraire calculé qui contourne les zones déconseillées dans lesquelles la congestion (due aux travaux, zone non déneigée, etc.) a été détectée au cours des expériences. Nous avons également évalué le coût supplémentaire encouru par TransportML, dû principalement à l'appel des services, et de la communication sans fil.

La plateforme TransportML est dynamique et flexible, ce qui permet de rajouter de nouveaux services à la communauté et de bénéficier des informations échangées. De même, la plateforme est extensible, car aucun scénario prédéfini n'est nécessaire au moment du développement. Cependant, pour un scénario donné, lorsqu'un ensemble de services fournisse le même résultat, le choix entre les services pour réaliser une tâche est effectué en se basant sur les critères de la QoS de chacun des services candidats. Nous avons donc étudié le comportement des services lors de l'appel de ces derniers, et les résultats montrent l'importance de la prise en compte de ces critères dans l'élaboration d'un scénario de composition de services.

Dans le chapitre suivant, nous allons présenter le bilan ainsi que les perspectives à nos travaux de recherche présentés dans cette thèse. Nous allons discuter des points à améliorer et des évolutions futures de nos approches.

Chapitre VI

Conclusion générale et perspectives

1 Bilan

Dans ce mémoire, une approche orientée modèle pour la vérification et l'évaluation de performances de l'interopérabilité et l'interaction de service est proposée. La composition de services est principalement basée sur l'intégration des workflows patterns. En effet, nous avons développé une sémantique formelle de certains patterns potentiels participants à la composition de services. Plus précisément, nous avons proposé des modèles mathématiques pour décrire le comportement analytique des différents patterns participant à la composition de services. La sémantique formelle proposée fournit une correspondance directe entre chaque pattern à son modèle (max, +), *via* un modèle de Réseau de Petri P-temporisé décrivant le comportement graphique de la configuration du pattern.

Ce travail de recherche s'inscrit dans le cadre des deux projets Européens ASSET et TeleFOT dont nous sommes impliqués et qui se focalisent sur la sécurité routière dans le domaine du transport. Dans ce cadre, nous avons initié et développé une plateforme d'interopérabilité de services à base de positionnement. Dans la composition automatique, où les services doivent être bien décrits et leurs capacités doivent être bien définies, ni les services impliqués ni le scénario de composition ne sont identifiés lors de l'étape de développement. En conséquence, le scénario est établie à la volée pour satisfaire la requête d'un utilisateur ainsi, les services impliqués dans ce scénario sont choisis et sélectionnés au moment de l'exécution. Le service de sélection est basé sur divers critères de la QoS définis à partir de la demande de l'utilisateur. Un nouveau langage, appelé TML, a été définit pour permettre cette composition automatique de service.

Par le biais d'un ensemble de propriétés quantitatives et qualitatives vérifiée à l'aide des modèles (max, +) et RdP, la faisabilité et l'expressivité de la composition de services sont garanties. L'approche et la méthodologie proposées ont été illustrées par une étude de cas. Dans

cette étude, nous avons proposé un scénario de composition de services de secours pour une intervention d'urgence. Nous avons montré, à travers cette étude, la faisabilité de l'approche proposée et sa capacité d'évaluer les performances des services composés. Les modèles développés et les résultats relatifs à l'analyse et l'évaluation de performance, ainsi qu'à l'implémentation des processus de composition de service ont été exposés et commentés.

L'évaluation des performances de la plateforme TransportML a été réalisée grâce à des expériences réelles en collaboration avec les autorités locales de la ville de Belfort à la fois dans des zones urbaines et sur l'autoroute. La QoS a été introduite, comme critère de sélection, afin de permettre une composition pertinente et optimale de services.

Bien que le prototype a été réalisé et testé avec succès, certaines fonctionnalités initialement prévues pour la plateforme TransportML ne sont pas encore implémentées. En conséquence, le prototype n'a pas encore atteint le potentiel souhaité de la plateforme et des travaux sont en cours pour amener le prototype à un prochain jalon. Ces perspectives ainsi qu'une extension de la modélisation et l'évaluation formelles sont décrites dans la section suivante.

2 Perspectives

Nos travaux futurs dans le domaine ainsi que les perspectives de ce travail peuvent être déclinés de la manière suivante. Une première perspective consiste à automatiser et générer les modèles associés aux scénarios de composition de services. En effet, à partir d'un scénario décrit en utilisant les workflow patterns, nous générons le modèle (max,+) associé. Nous généraliserons également notre approche sur d'autres workflow patterns. Nous allons également étudier le modèle de façon globale en prenant en compte la réception simultanée de plusieurs requêtes afin que les modèles se rapprochent des situations concrètes. En outre, d'autres critères de qualité de service seront prises en compte et intégrés dans les modèles formels en vue d'assurer une meilleure sélection de services avec des meilleurs scores de qualité de service. D'autres expériences et tests, à grande échelle, seront effectués en prenant en compte d'autres métriques d'évaluation (d'autres critères de QoS).

Une autre perspective que nous avons identifiée est relative à la composition dynamique. En effet, au lieu d'utiliser des documents TML pour décrire les services Web, il est possible d'obtenir de meilleurs résultats en utilisant des technologies sémantiques. Ces technologies utilisent des langages tels que OWL (Web Ontology Language) et des ontologies pour produire des descriptions très précises et exhaustives qui peuvent être comprises par les machines. En utilisant ces technologies, la plateforme TransportML sera capable de faire des choix plus précis dans la

découverte et la sélection de services. Cela se traduira par des réponses plus pertinentes aux requêtes des utilisateurs. L'extensibilité des ontologies nous permettra de faire évoluer les langages de description de services et ainsi ajouter de nouvelles propriétés ou valeurs aux descriptions des services Web. Ceci est important car les besoins des utilisateurs et des applications qui interagissent avec la plateforme TransportML sont susceptibles d'évoluer. L'utilisation des descriptions sémantiques pour les services affecteront également d'autres tâches que la découverte et la sélection de services. En effet, ces descriptions permettront également des améliorations dans l'élaboration des scénarios de la composition de services grâce à une meilleure gestion des dépendances interservices.

Une autre perspective de la plateforme consiste à étudier son évolution dans le cas où plusieurs requêtes seront exécutées en même temps. En effet, la plateforme pourrait recevoir un grand nombre de demandes des utilisateurs. Dans un tel contexte, les temps de réponse deviennent anormalement long ce qui pourrait mener à un impact indésirable notamment dans le cas des requêtes établies par des services de secours d'urgence. En conséquence, les performances de la plateforme sont un point critique qu'il faudrait prendre en considération, d'une part, en optimisant toutes les étapes et opérations liées aux traitements des requêtes, et d'autre part, en répartissant les moteurs TransportML (appelés aussi *Broker*) sur des zones de couverture définies. En d'autres termes, chaque *Broker* couvrira une zone et traitera seules les requêtes liées à cette zone. Cette solution est principalement une conséquence de l'utilisation des LBS qui dépendent souvent de la localisation des mobiles (p. ex. véhicules). Pour cette raison, les LBS ont souvent des zones de couverture, ce qui signifie qu'ils ne répondent qu'aux demandes émises par les véhicules ou requêtes dans leurs zones de couverture. Par conséquent, les fournisseurs des LBS doivent enregistrer leurs services dans des annuaires géographiques en étendant la structure UDDI actuelle de TransportML, et fournir des informations sur leurs zones de couverture. Pour atteindre une telle architecture, des serveurs TransportML sont nécessaires, ce que l'on appelle les registres spatiaux. Leur fonctionnalité est incluse dans leur nom ; les *brokers* TransportML doivent s'inscrire dans ces serveurs et fournir des informations sur leurs zones de couverture. De cette façon, lors de la communication des véhicules ou mobiles avec ces serveurs, ces derniers vont affecter ces mobiles au *broker* de leur zone. Il est probable dans ce cas, que les mobiles se déplacent hors de la zone de couverture de leur broker TransportML et entrent dans une zone de couverture d'un autre broker. C'est ce que l'on appelle *HandOver* (HO). En effet des problèmes similaires ont été rencontrés et résolus dans les réseaux cellulaires. La plateforme TransportML devrait traiter ces HO de la même façon en faisant usage des registres spatiaux.

Finalement, nous allons également améliorer les performances de la plateforme en réduisant le temps nécessaire pour réaliser les différentes opérations pour traiter une requête. Nous envisa-

geons, par exemple, d'utiliser le cache pour répondre aux requêtes du même type. TransportML pourrait gérer le cache, de la même manière que les serveurs Web HTTP. Il est probable que la plateforme doive répondre à des demandes identiques ou très similaires dans un temps très court. Elle doit également se souvenir de la réponse aux demandes antérieures dans un cache et envoyer cette réponse précédemment calculée. Ceci permettra de réduire la charge de la plateforme et des services impliqués. Selon le type des demandes, le cache peut être un moyen très efficace pour réduire les temps de réponse. En effet, certaines réponses ne sont pas susceptibles d'évoluer au fil du temps (p. ex. l'emplacement de restaurants dans une zone donnée). De plus, introduire un second type de cache pourrait apporter beaucoup d'avantages à la plateforme. En effet, chaque invocation de service nécessite de récupérer et d'analyser le WSDL afin de générer une requête SOAP valide. Sachant que les documents WSDL ne sont pas souvent modifiés au fil du temps, puisque l'interface d'un service n'évolue pas beaucoup après sa publication, la mis en cache des documents WSDL pourrait également réduire les temps de réponse, d'une façon significative, en obtenant une meilleure efficacité en termes d'invocation de services.

Publications

1. Revues internationales

[1] W. Ait-Cheik-Bihi, A. Nait-Sidi-Moh, M. Bakhouya, J. Gaber et M. Wack. TransportML platform for collaborative Location-based services. Á apparaître dans le journal *Service Oriented Computing and Applications* (SOCA 2012).

[2] W. Ait-Cheik-Bihi, A. Nait-Sidi-Moh, M. Bakhouya, J. Gaber et M. Wack. Un Système Coopératif Routier basé sur des Techniques du Geofencing et la Communication sans Fils pour l'Aide à la Conduite. Soumis au journal Recherche Transport et Sécurité (RTS), 2012.

2. Conférences internationales avec actes

[1] W. Ait-Cheik-Bihi, A. Nait-Sidi-Moh, M. Bakhouya, J. Gaber et M. Wack. Performance study of workflow patterns-based Web service composition. Á apparaître dans les actes de 9th International Conference on Mobile Web Information Systems (MoibWIS 2012).

[2] W. Ait-Cheik-Bihi, M Bakhouya, A Nait-Sidi-Moh, J Gaber et M Wack. A Platform for Interactive Location-Based Services. In 8th International Conference on Mobile Web Information System MobiWIS'11, September 19-21, 2011, Niagara Falls, Ontario, Canada. Procedia Computer science, Volume 5, 2011, Pages 697-704.

[3] W. Ait-Cheik-Bihi, A Chariette, M Bakhouya, A Nait-Sidi-Moh, J Gaber et M Wack. An

In-vehicle Emergency Call Platform for Efficient Road Safety. Dans les actes de 8th ITS European Congress, June 6-9, 2011, Lyon, France.

[4] W Ait-Cheik-Bihi, A Nait-Sidi-Moh et M Wack. Conflict Management and Resolution Using (Max, +) Algebra : Application to Services Interaction. In 8th International Conference of Modeling and Simulation MOSIM'10, May 10-12, 2010, Hammamet, Tunisie.

[5] A Nait-Sidi-Moh, W Ait-Cheik-Bihi et M Wack. Modeling and Performance Analysis of a Bus Network Using Petri Nets and Dioid Algebra. In 39th International Conference on Computers and Industrial Engineering (CIE39), July 6-8, 2009, Troyes, France.

Table des figures

Liste des tableaux

Glossaire

BPEL	Business Process Execution Language, 92
BPM	Business Process Management, 35
CADP	Construction and Analysis of Distributed Processes, 38
CCS	Calculus of Communicating Systems, 37
CSP	Calculus of Sequential Processes, 37
GET	Graphe d'Événements Temporisés, 47
GML	Geography Markup Language, 80
GNSS	Global Navigation Satellite System, 77
GPRS	General Packet Radio Service, 78
LOTOS	Language Of Temporal Ordered Systems, 37
MDE	Model Driven Engineering, 16
P2P	l'acronyme en anglais de peer-to-peer, qui est un modèle de réseau informatique proche du modèle client-serveur mais où chaque client est aussi un serveur, 23
PDA	Personal Digital Assistant, 77

QoS	Quality of Service, 86
SOA	Architecture Orientée Service, 7
SOC	Service-Oriented Computing, 45
TML	Tranport Markup Language, 80
TML SD	TransportML Service Description, 88
UDDI	Universal Description Discovery and Integration, 8

Références bibliographiques

[A.04] White Stephen A. Business process modeling notation (bpmn) version 1.0. Technical report, 2004. 34

[ACBBNSM+11] W Ait-Cheik-Bihi, M Bakhouya, A Nait-Sidi-Moh, J Gaber, and M Wack. A platform for interactive location-based services. In *8th International Conference on Mobile Web Information Systems, Procedia Computer Science, Volume 5 697-704, Elsevier. September 2011, Niagara Falls, Ontario, Canada.*, volume 5, 2011. 113

[ACBCB+11] W Ait-Cheik-Bihi, A Chariette, M Bakhouya, A Nait-Sidi-Moh, J Gaber, and M Wack. An in-vehicle emergency call platform for efficient road safety. In *8th ITS Europeen congress, Jun 2011, Lyon, France.*, June 6 2011. 3, 78, 111, 112, 113

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126 :183–235, 1994. 36

[AGGI04] Jim Amsden, Tracy Gardner, Catherine Griffin, and Sridhar Iyengar. Draft UML 1.4 profile for automated business processes with a mapping to BPEL 1.0. Technical report, IBM, July 2004. 33

[AIS77] Christopher Alexander, Sara Ishikawa, and Murray Silverstein. *A Pattern Language : Towns, Buildings, Construction (Center for Environmental Structure Series)*. Oxford University Press, later printing edition, August 1977. 43

[BB87] Tommaso Bolognesi and Ed Brinksma. Introduction to the iso specification language lotos. *Comput. Netw. ISDN Syst.*, 14 :25–59, March 1987. 37

[BBG07] Maurice H Beek, Antonio Bucchiarone, and Stefania Gnesi. Formal methods for service composition. *Annals of Mathematics Computing Teleinformatics*, 1(5) :1–14, 2007. 37

[BG07] Mohamed Bakhouya and Jaafar Gaber. Service composition approaches for ubiquitous and pervasive computing environments : A survey. In Eldon Li and Soe-Tsyr Yuan, editors, *Agent Systems in Electronic Business*, pages 323–350. Information Science Reference/IGI Publishing, 2007. 21

[BGB05] Reda Bendraou, Marie-Pierre Gervais, and Xavier Blanc. Uml4spm : a uml2.0-based metamodel for software process modelling. In *Proceedings of the 8th international conference on Model Driven Engineering Languages and Systems*, MoDELS'05, pages 17–38, Berlin, Heidelberg, 2005. Springer-Verlag. 33

[BHLJ04] Jean Bezivin, Slimane Hammoudi, Denivaldo Lopes, and Jouault Jouault. Applying mda approach for web service platform. In *Proceedings of the Enterprise Distributed Object Computing Conference, Eighth IEEE International*, pages 58–70, Washington, DC, USA, 2004. IEEE Computer Society. 19

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. 2001. 26

[BPG03] Rajive Bagrodia, Thomas Phan, and Richard Guy. A scalable, distributed middleware service architecture to support mobile internet applications. *Wirel. Netw.*, 9(4) :311–320, July 2003. 78

[BPG06] Sami Bhiri, Olivier Perrin, and Claude Godart. Extending workflow patterns with transactional dependencies to define reliable composite web services. *Advanced International Conference on Telecommunications / Internet and Web Applications and Services, International Conference on*, 0 :145, 2006. 46

[Bur06] David S. Burggraf. Geography markup language. *Data Science Journal*, 5 :178–204, 2006. 14

[Cai11] Shichao Cai. *Systéme d'information et d'assistance à la conduite pour la mise en oeuvre de la sécurité à bord d'un véhicule*. PhD thesis, UTBM, Novembre 2011. 109

[cbsmW10] W. Ait cheik bihi, A. Nait sidi moh, and M. Wack. Conflict management and resolution using (max, +) algebra : Application to services interaction. In *The 8th International Conference of Modeling and Simulation (MOSIM'10)*, 2010. 38, 62

[CCS08] Shuo-Yan Chou, Yao-Hui Chang, and Chun-Ying Shen. A fuzzy simple additive weighting system under group decision-making for facility location selection with objective/subjective attributes. *European Journal of Operational Research*, 189(1) :132–145, 2008. 49

[Cha02] Jean-Marie Chauvet. *Service Web avec SOAP, WSDL,UDDI, ebXML...* Eyrolles edition, mars 2002. 8

[CM89] Guy Cohen and Pierre Moller. Algebraic tools for the performance evaluation of discrete event systems. In *IEEE Proceedings : Special issue on Discrete Event Systems*, pages 39–58, 1989. 39, 52, 57

[DAM11] Daml : Daml-s (and owl-s) 0.9 draft release from http ://www.daml.org/services/daml-s/0.9/, 2011. 27

[DBG+10] K Dar, M Bakhouya, J Gaber, M Wack, and P Lorenz. Wireless communication technologies for its applications. *IEEE Communications Magazine*, 48(5) :156–162, 2010. 13, 77

[DGW08] Christophe Dumez, Jaafar Gaber, and Maxime Wack. Model-driven engineering of composite web services using uml-s. In *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services*, iiWAS '08, pages 395–398, New York, NY, USA, 2008. ACM. 19, 33, 38, 45

[DKL+05] Asuman Dogac, Yildiray Kabak, Gokce B. Laleci, Carl Mattocks, Farrukh Najmi, and Jeff Pollock. Enhancing ebxml registries to make them owl aware. *Distrib. Parallel Databases*, 18(1) :9–36, 2005. 27

[DPC+05] Gregorio Díaz, Juan José Pardo, María-Emilia Cambronero, Valentin Valero, and Fernando Cuartero. Automatic translation of ws-cdl choreographies to timed automata. In Mario Bravetti, Leïla Kloul, and Gianluigi Zavattaro, editors, *EPEW/WS-FM*, volume 3670 of *Lecture Notes in Computer Science*, pages 230–242. Springer, 2005. 37

[DS05] Schahram Dustdar and Wolfgang Schreiner. A survey on web services composition. *IJWGS*, 1(1) :1–30, 2005. 20, 27

[Dua11] Qiang Duan. Network service description and discovery for high-performance ubiquitous and pervasive grids. *ACM Trans. Auton. Adapt. Syst.*, 6 :3 :1–3 :17, February 2011. 38

[Dum10] Christophe Dumez. *Approche dirigée par les modèles pour la spécification, la vérification formelle et la mise en oeuvre de services Web composés.* PhD thesis, UTBM, Août 2010. 37

[Erl07] Thomas Erl. *SOA Principles of Service Design.* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2007. 7

[fDSC11] OWL-S (formerly DAML-S) Coalition. Owl : Owl-s 1.2 release from http ://www.ai.sri.com/daml/services/owl-s/1.2/, 2011. 27

[Fen03] Dieter Fensel. *Ontologies : A Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag, Heidelberg, 2003. 27

[Fer04] Andrea Ferrara. Web services : a process algebra approach. In *Proceedings of the 2nd international conference on Service oriented computing*, ICSOC '04, pages 242–251, New York, NY, USA, 2004. ACM. 38

[FGGJJ92] Baccelli F, Cohen G, Olsder G-J, and Quadrat JP. Synchronisation and linearity, algebra for discrete event systems, 1992. 39, 55, 57

[FMS+05] Keita Fujii, Student Member, Tatsuya Suda, A. Dynamic, and Service Composition. Semantics-based dynamic service composition, to appear in the. *IEEE Journal on Selected Areas in Communications (JSAC), special issue on Autonomic Communication Systems*, 23 :2361–2372, 2005. 27

[GHM+11] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, Anish Karmarkar, and Yves Lafon. Soap version 1.2 part 1 : Messaging framework (second edition), 2011. 9

[Gro08] Object Management Group. Business processmodeling notation, v1.1 omg available specification, January 2008. 33

[HB03] Rachid Hamadi and Boualem Benatallah. A petri net-based model for web service composition. In *Proceedings of the 14th Australasian database conference - Volume 17*, ADC '03, pages 191–200, Darlinghurst, Australia, Australia, 2003. Australian Computer Society, Inc. 35

[HBL06] S. Hamaci, J.-L. Boimond, and S. Lahaye. Modeling and control of hybrid timed event graphs with multipliers using (min, +) algebra. *Discrete Event Dynamic Systems*, 16 :241–256, 2006. 10.1007/s10626-006-8135-7. 39, 56

[Hoa83] C. A. R. Hoare. Communicating sequential processes. *Commun. ACM*, 26 :100–106, January 1983. 37

[HOvdW05] Bernd Heidergott, Geert J. Olsder, and Jacob van der Woude. *Max Plus at Work : Modeling and Analysis of Synchronized Systems : A Course on Max-Plus Algebra and Its Applications (Princeton Series in Applied Mathematics)*. Princeton University Press, November 2005. 39, 56

[HR04] Thomas Hadig and Jörg Roth. Proximity services with the nimbus framework. In *IADIS International Conference Applied Computing 2004*, pages 23–26. IADIS Press, 2004. 78

[HSS05] Sebastian Hinz, Karsten Schmidt, and Christian Stahl. Transforming bpel to petri nets. In *Proceedings of the International Conference on Business Process Management (BPM2005), volume 3649 of Lecture Notes in Computer Science*, pages 220–235. Springer-Verlag, 2005. 35

[Hu03] Michael Hu. Web services composition, partition, and quality of service. In *in Distributed System Integration and Reengineering, presented at XML Conference 2003*, 2003. 22

[HYJY08] Qiang He, Jun Yan, Hai Jin, and Yun Yang. Adaptation of web service composition based on workflow patterns. In *ICSOC*, pages 22–37, 2008. 45, 46

[Jen03] Kurt Jensen. *Coloured Petri Nets : Basic Concepts, Analysis Methods and Practical Use (Volume 1)*, volume 1 of *EATCS Series*. Springer Verlag, 2003. 44

[JMJM06] FAVRE Jean-Marie, ESTUBLIER Jacky, and BLAY-FORNARINO Mireille. *L'ingénierie dirigée par les modèles. Au-delà du MDA*. Hermès - Lavoisier, 2006. 17, 19

[JMW+07] Kurt Jensen, Lars Michael, Kristensen Lisa Wells, K. Jensen, and L. M. Kristensen. Coloured petri nets and cpn tools for modelling and validation of concurrent systems. In *International Journal on Software Tools for Technology Transfer*, page 2007, 2007. 35

[JRGM04] M. C. Jaeger, G. Rojec-Goldmann, and G. Muhl. QoS aggregation for web service composition using workflow patterns. In *Proceedings. Eighth IEEE International Enterprise Distributed Object Computing Conference, 2004. EDOC 2004.*, pages 149–159. IEEE, 2004. 45

[KBR+05] Nickolas Kavantzas, David Burdett, Gregory Ritzinger, Tony Fletcher, Yves Lafon, and Charlton Barreto. Web services choreography description language version 1.0. World Wide Web Consortium. From http ://www.w3.org/TR/2005/CR-ws-cdl-10-20051109/, 2005. 25

[KC10] Mehmet Kuzu and Nihan Cicekli. Dynamic planning approach to automated web service composition. *Applied Intelligence*, June 2010. 27

[KL03] Hubert Ka and Yau Leung. Modeling location-based services with subject spaces. In *Proceedings of the 2003 conference of the Centre for Advanced Studies on Collaborative Research*, page 171, 2003. 78

[KLB09] J. KOMENDA, S. LAHAYE, and J.-L. BOIMOND. Le produit synchrone des automates (max,+). *Special issue of Journal Européen des Systèmes Automatisés (JESA)*, 43/7-9 :1033–1047, 2009. 36

[LBT01] Jean-Yves Le Boudec and Patrick Thiran. *Network calculus : a theory of deterministic queuing systems for the internet*. Springer-Verlag, Berlin, Heidelberg, 2001. 38

[LDBL07] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 37(6) :1067–1080, November 2007. 13

[LJL+11] KangChan Lee, JongHong Jeo, WonSeok Lee, Seong-Ho Jeong, and Sang-Won Park. Qos for web services : Requirements and possible approaches, 2011. 28

[LNZ04] Yutu Liu, Anne H. Ngu, and Liang Z. Zeng. Qos computation and policing in dynamic web service selection. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, WWW Alt. '04, pages 66–73, New York, NY, USA, 2004. ACM. 28

[LPY97] Kim G. Larsen, Paul Pettersson, and Wang Yi. Uppaal in a nutshell. *Int. Journal on Software Tools for Technology Transfer*, 1 :134–152, 1997. 36

[LT87] Nancy A. Lynch and Mark R. Tuttle. Hierarchical correctness proofs for distributed algorithms. Technical report, 1987. 36

[LT89] Nancy A. Lynch and Mark R. Tuttle. An introduction to input/output automata. *CWI Quarterly*, 2 :219–246, 1989. 36

[MGI06] S. B. Mokhtar, N. Georgantas, and V. Issarny. COCOA : Conversation-Based service composition for pervasive computing environments. In *2006 ACS/IEEE International Conference on Pervasive Services*, pages 29–38. IEEE, 2006. 37

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989. 37

[MM03] J. Miller and J. Mukerji. Mda guide version 1.0.1. Technical Report omg/03-06-01, Object Management Group (OMG), June 2003. 17

[NM02a] Srini Narayanan and Sheila A. McIlraith. Simulation, verification and automated composition of web services. In *Proceedings of the 11th international conference on World Wide Web*, WWW '02, pages 77–88, New York, NY, USA, 2002. ACM. 35

[NM02b] Srini Narayanan and Sheila A. Mcilraith. Simulation, verification and automated composition of web services. In *World Wide Web Conference Series*, pages 77–88, 2002. 35

[NSMACBW09] A Nait-Sidi-Moh, W Ait-Cheik-Bihi, and M Wack. Modelling and analysis of a non-synchronized transport network using petri nets and (max, plus) algebra. In *39th International Conference on Computers ; Industrial Engineering (CIE39), Troyes, France.*, july 2009. 39, 54

[ODtHvdA07] Chun Ouyang, Marlon Dumas, Arthur H.M. ter Hofstede, and Wil M.P. van der Aalst. Pattern-based translation of bpmn process models to bpel web services. *International Journal of Web Services Research (JWSR)*, 5(1) :42–62, 2007. 34

[Oli05] Ian Oliver. Applying uml and mda to real systems design. In *Proceedings of the conference on Design, Automation and Test in Europe - Volume 1*, DATE '05, pages 70–71, Washington, DC, USA, 2005. IEEE Computer Society. 19

[OVB+05] Chun Ouyang, Eric Verbeek, Stephan Breutel, Marlon Dumas, and Arthur H. M. Ter Hofstede. Wofbpel : A tool for automated analysis of bpel processes. In *ICSOC 2005 proceedings, December, Lecture Notes in Computer Science*, pages 484–489. Springer, 2005. 51

[OVvdA+07] Chun Ouyang, Eric Verbeek, Wil M. P. van der Aalst, Stephan Breutel, Marlon Dumas, and Arthur H. M. ter Hofstede. Formal semantics and analysis of control flow in ws-bpel. *Sci. Comput. Program.*, 67 :162–198, July 2007. 35

[Pet62] C. A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut für instrumentelle Mathematik, Bonn, 1962. 35

[PFC08] Jose Antonio Parejo, Pablo Fernandez, and Antonio Ruiz Cortés. Qos-aware services composition using tabu search and hybrid genetic algorithms. *Computer Languages*, 2(1) :55–66, 2008. 28, 46, 47

[PZWQ06] Geguang Pu, Xiangpeng Zhao, Shuling Wang, and Zongyan Qiu. Towards the semantics and verification of bpel4ws. *Electron. Notes Theor. Comput. Sci.*, 151 :33–52, May 2006. 37

[QXQY09] Fang Qiqing, Peng Xiaoming, Liu Qinghua, and Hu Yahui. A global qos optimizing web services selection algorithm based on moaco for dynamic web service composition. volume 1, pages 37 –42, may. 2009. 26, 28, 46

[RA05] E.E. Roubtsova and M. Aksit. Extension of petri nets by aspects to apply the model driven architecture approach. In *Preliminary Proceedings of the 1st International Workshop on Aspect-Based and Model-Based Separation of Concerns in Software Systems (ABMB)*, 2005. 19, 20

[RAvdAM06] Nick Russell, Arthur, Wil M. P. van der Aalst, and Natalya Mulyar. Workflow Control-Flow patterns : A revised view. Technical report, BPMcenter.org, 2006. 44, 45

[REC09] Fabrice RECLUS. *Géopositionnement et mobilités : GPS, EGNOS et GALILEO*, chapter Geofencing, pages 297–321. 2009. 102

[RGWNSM07] A Roxin, J Gaber, M Wack, and A Nait-Sidi-Moh. Survey of wireless geolocation techniques. In *proceedings of IEEE-GLOBECOM, Workshop SUPE, 26-30 November 2007, Washington , DC, USA. CD-ROM : ISBN 1-4244-1943-6.*, 2007. 12, 13

[SBS04] Gwen Salaün, Lucas Bordeaux, and Marco Schaerf. Describing and reasoning on web services using process algebra. In *Proceedings of the IEEE International Conference on Web Services*, ICWS '04, pages 43–, Washington, DC, USA, 2004. IEEE Computer Society. 37

[smMMM03] A Nait sidi moh, M A Manier, A El Moudni, and H Manier. Performance analysis of a bus network based on petri nets and (max, +) algebra. *International Journal of Systems Science (ex. SAMS : Systems Analysis Modelling Simulation), Editeur Taylor 38 ; Francis, ISSN : 0020-7721*, 43(5) :639–669, May 2003. 54

[SV90] Manuel Silva and Robert Valette. Petri nets and flexible manufacturing. In *Advances in Petri Nets 1989, covers the 9th European Workshop on Applications and Theory in Petri Nets-selected papers*, pages 374–417, London, UK, UK, 1990. Springer-Verlag. 52

[SZS10] Kawther Saeedi, Liping Zhao, and Pedro R. Falcone Sampaio. Extending bpmn for supporting customer-facing service quality requirements. In *Proceedings of the 2010 IEEE International Conference on Web Services*, ICWS '10, pages 616–623, Washington, DC, USA, 2010. IEEE Computer Society. 34

[tea10] GSEM team. Asset-del5.2 : Specification and implantation of geofencing for security applications. internal deliverable, UTBM, june 2010. 102

[TVM+03] Aphrodite Tsalgatidou, Jari Veijalainen, Jouni Markkula, Artem Katasonov, and Stathes Hadjiefthymiades. Mobile e-commerce and location-based services : Technology and requirements. In *ScanGIS*, pages 1–14, 2003. 11

[vdA96] W. M. P. van der Aalst. Three good reasons for using a Petri-net-based workflow management system. In S. Navathe and T. Wakayama, editors, *Proceedings of the International Working Conference on Information and Process Integration in Enterprises (IPIC'96),*, pages 179–201, 1996. 45, 51, 52

[vdA98] Wil M. P. van der Aalst. The application of petri nets to workflow management. *Journal of Circuits, Systems, and Computers*, 8(1) :21–66, 1998. 35, 58

[vdAHKB00] W. M. P. van der Aalst, Ter A. H. M. Hofstede, B. Kiepuszewski, and A. P. Barros. Advanced workflow patterns. In *Proceedings of the 7th International Conference on Cooperative Information Systems (CoopIS 2000)*, volume 1901 of *Lecture Notes in Computer Science*, pages 18–29, 2000. 44

[Vir01] *Developing GIS-supported location-based services*, volume 2, 2001. 79

[Whi04] Stephen A. White. Process modeling notations and workflow patterns. On BPMN website, 2004. 34

[WNSMG+09] M Wack, A Nait-Sidi-Moh, J Gaber, P-Y Gillieron, and Y Alexandre. *Géo-positionnement et mobilités : GPS, EGNOS et GALILEO.* 2009. 12

[WvdAP+03] Petia Wohed, Wil M.P. van der Aalst, Wil M. P, Marlon Dumas, and Arthur H.M. ter Hofstede. Analysis of web services composition languages : The case of bpel4ws. In *Proc. of ER'03, LNCS 2813*, pages 200–215. Springer Verlag, 2003. 35

[YGN09] Ustun Yildiz, Adnene Guabtni, and Anne H. H. Ngu. Towards scientific workflow patterns. In *Proceedings of the 4th Workshop on Workflows in Support of Large-Scale Science*, WORKS '09, pages 13 :1–13 :10, New York, NY, USA, 2009. ACM. 44

[YK04a] Xiaochuan Yi and Krys J. Kochut. A cp-nets-based design and verification framework for web services composition. In *Proceedings of the IEEE International Conference on Web Services*, ICWS '04, pages 756–, Washington, DC, USA, 2004. IEEE Computer Society. 35

[YK04b] Xiaochuan Yi and Krys J. Kochut. Process composition of web services with complex conversation protocols : a colored petri nets based approach. Proceedings of the Design, Analysis, and Simulation of Distributed Systems, 2004. 52

[YM08] Xinfeng Ye and Rami Mounla. A hybrid approach to qos-aware service composition. In *ICWS*, pages 62–69, 2008. 28

[Yov97] Sergio Yovine. Kronos : A verification tool for real-time systems. (kronos user's manual release 2.2). *International Journal on Software Tools for Technology Transfer*, 1 :123–133, 1997. 36

[ZBD+03] Liangzhao Zeng, Boualem Benatallah, Marlon Dumas, Jayant Kalagnanam, and Quan Z. Sheng. Quality driven web services composition. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 411–421, New York, NY, USA, 2003. ACM. 28

Résumé

De nos jours, les services Web sont très utilisés notamment par les entreprises pour rendre accessibles leurs métiers, leurs données et leurs savoir-faire via le Web. L'émergence des services Web a permis aux applications d'être présentées comme un ensemble de services métiers bien structurés et correctement décrits, plutôt que comme un ensemble d'objets et de méthodes. La composition automatique de services est une tâche complexe mais qui rend les services interopérables, ainsi leur interaction permet d'offrir une valeur ajoutée dans le traitement des requêtes des utilisateurs en prenant en compte des critères fonctionnels et non fonctionnels de la qualité de service. Dans ce travail de thèse, nous nous intéressons plus précisément aux services à base de localisation (LBS) qui permettent d'intégrer des informations géographiques, et de fournir des informations accessibles depuis des appareils mobiles via, les réseaux mobiles en faisant usage des positions géographiques de ces appareils.
L'objectif de ce travail est de proposer une approche orientée modèles pour spécifier, valider et mettre en oeuvre des processus de composition automatique de services à des fins de sécurité routière dans les transports. Cette approche est basée sur deux outils formels à savoir les Réseaux de Petri (RdP) et l'algèbre (max,+). Pour cela, nous préconisons l'utilisation des workflow patterns dans la composition, où chaque pattern est traduit par un modèle RdP et ensuite par une équation mathématique dans l'algèbre (max,+). Les modèles formels développés ont conduit, d'une part, à la description graphique et analytique des processus considérés, et d'autre part, à l'évaluation et la vérification quantitatives et qualitatives de ces processus. Une plateforme, appelée TransportML, pour la collaboration et l'interopérabilité de services à base de positionnement a été implémentée. Les résultats obtenus par la simulation des modèles formels sont comparés à ceux issus des simulations du fonctionnement de la plateforme et des expérimentations sur le terrain.
Cette thèse est effectuée dans le cadre des projets Européens FP7 ASSET (2008-2011) et TeleFOT (2008-2012).

Mots clés : Composition de services, Service à base de localisation, Architecture SOA, Modélisation, Évaluation de performance, Vérification formelle, Réseaux de Petri, Algèbre $(max, +)$.

Abstract

Web services are widely used by organizations to share their knowledge over the network and facilitate business-to-business collaboration. The emergence of Web services enabled applications to be presented as a set of business services well structured and correctly described. However, combining Web services and making them interoperable, to satisfy user requests taking into account functional and non-functional quality criteria, is a complex process. In this work, we focus specifically on location-based services (LBS) that integrate geographic information and provide information reachable from mobile devices, through wireless network by making use of the geographical positions of the devices.
The aim of this work is to develop a model driven approach to specify, validate and implement service composition process in an automatic fashion for road security. This approach is based on two formal tools namely Petri nets (PN) and (max, +) algebra used to model, to verify and to evaluate the performance of service composition process. Workflow patterns are used to represent service composition processes. The behavior of each pattern is modeled by a PN model and then by a (max,+) state equation. The developed formal models allow the graphical and analytical description of the considered processes. Also, these models enable to evaluate some quantitative and qualitative properties of the considered processes. A platform, called TransportML, has been developed for collaboration and interoperability of different LBS. The obtained simulation results from the formal models are compared, on one hand, to those obtained from trials of the platform, and on the other hand, to those obtained from the real experimentations on the field.
This work is a part of the FP7 European projects ASSET (2008-2011) and TeleFOT (2008-2012).

Keywords : Service composition, Location-Based Service (LBS), Service-Oriented Architecture (SOA), Modeling and Performance evaluation, Formal verification, Petri Nets, $(max, +)$ algebra.