Architecture multi-agents pour la composition automatique de web services

Julien Bourdon* — **Philippe Beaune*** — **Humbert Fiorino****

* *Centre G2I - Equipe SMA*
Ecole Nationale Supérieure des Mines de Saint-Etienne
158, cours Fauriel F-42023 SAINT-ÉTIENNE cedex 2
{bourdon,beaune}@emse.fr

** *LIG (UMR 5217) - Equipe Magma*
46, avenue Félix Viallet
F - 38031 Grenoble cedex
Humbert.Fiorino@imag.fr

RÉSUMÉ. *Les propositions actuelles de composition automatique de web services atteignent leurs limites dans le cas d'environnements distribués et dynamiques. Nous proposons dans cet article de résoudre ce problème en utilisant la planification distribuée multi-agents. Les agents de notre architecture génèrent collaborativement une description exécutable de service composite, prenant en compte la QoS.*

ABSTRACT. *The existing web service automatic composition approaches are not scaled to dynamic and distributed environments. In this paper, we thus propose to use multi-agent distributed planning. The agents of our architecture collaborate to synthesise an executable description of a QoS-aware composite service.*

MOTS-CLÉS : *Composition automatique de web services, planification distribuée multi-agents, QoS*

KEYWORDS: *Automatic composition of web services, distributed multi-agent planning, QoS*

NOTE. — Ce travail est une action du projet Web Intelligence (`http://eric.univ-lyon2.fr/wi/`) soutenu par la Région Rhône-Alpes.

Introduction

Les architectures orientées services, notamment par le biais de web services, prennent peu à peu leur place dans les solutions B2B. Néanmoins, même si leur mise en place est déjà bien formalisée (Virdell, 2003), ces architectures ne sont pas adaptées aux contextes inter-organisationnels, distribués et dynamiques.

Dans ces contextes, la composition des web services doit être rendue automatique (Papazoglou *et al.*, 2006). De part leur adaptabilité aux environnements dynamiques, les approches multi-agents de la composition ont été largement étudiées. (Hendler, 2001) montre que la composition peut être réalisée par des agents, et (Singh *et al.*, 2005) propose de la considérer comme un problème de planification distribuée où un ensemble d'agents répartis sur plusieurs systèmes collaborent pour élaborer un plan, i.e. dans notre cas un service composite.

La première section introduit cette problématique et la deuxième section contient notre proposition d'architecture où la composition est résolue par un planificateur distribué.

1. De la composition à la planification distribuée

Afin de permettre une prise en charge automatique de la composition, il est essentiel de définir formellement la sémantique des services et des données qu'ils échangent par le biais d'ontologies. En ce qui concerne les processus décrits par les services, OWL-S (Martin *et al.*, 2004) une ontologie basée sur OWL permet de décrire ces services en terme de flux de contrôle, c'est-à-dire l'ordre dans lequel ils sont exécutés [1] ainsi qu'en terme de flux de données, c'est-à-dire comment leurs entrées et leurs sorties sont liées. D'autre part les préconditions, permettant l'exécution d'un service si son environnement est dans un certain état, et les effets, décrivant les changements qu'apporte un service à l'état de l'environnement, y sont aussi décrits.

Ordonner des web services ayant des préconditions et des effets est donc très similaire à un problème de planification automatique (Ghallab *et al.*, 2004). La planification a en effet pour rôle de trouver une séquence d'actions, ou plan, pour, à partir d'un état initial, arriver à un état but, exprimé par l'utilisateur. Un problème de planification est résolu dans le cadre d'un domaine, définissant les différentes actions possibles, leurs préconditions ainsi que leurs effets sur l'état du monde.

Le problème de la composition automatique de web services a déjà été étudié dans le projet ASTRO (Traverso *et al.*, 2004), dans la Language Grid (Ishida, 2006) et dans CASCOM (ADETTI, 2006). ASTRO propose un système transformant les services composés en systèmes à transition d'états. La planification est ensuite effectuée en utilisant des méthodes de *Model checking* et l'exécution peut être tracée pour voir si les contraintes de l'utilisateur ont été respectées. La Language Grid permet de repousser la sélection des services à l'exécution en utilisant le concept de services abstraits, rassemblant un panel de services concrets fonctionnellement équivalents ; les implémentations concrètes de ces services sont sélectionnées à partir de contraintes utili-

1. e.g. Séquence, branchement conditionnel, boucle,...

sateurs, notamment sur la QoS. Enfin CASCOM décrit une architecture totalement orientée agents et utilise une planification hybride *HTN/Forward planning* (Klusch *et al.*, 2005) pour générer un plan.

Néanmoins, tous ces systèmes proposent une planification centralisée, ce qui n'est pas adapté aux processus couvrant plusieurs organisations, possédant des informations hétérogènement décrites qui, de plus, peuvent être confidentielles.

Pour surmonter ces problèmes, une approche dialectique pour la synthèse de plans dans un contexte multi-agents a été proposée dans (Pellier, 2005). Un modèle de planification entièrement distribué où des agents raisonnent collectivement par un cycle de conjectures/réfutations pour construire un plan y est décrit. Néanmoins cette approche, qui a donné lieu à une implémentation nommée CoRe[2], n'est pas directement applicable à notre problématique : la description d'un problème de composition de web services diffère d'un problème de planification classique (Bourdon, 2007).

Nous proposons donc une architecture multi-agents adaptant les travaux décrits ci-dessus pour composer automatiquement des web services décrits sémantiquement tant au niveau des données qu'au niveau du processus. La sélection et le suivi pour la composition n'y sont pas traités même si la prise en considération de la QoS pourra le permettre dans un temps futur.

2. Des architectures orientées services aux architectures orientées agents

Dans cette section, nous décrivons notre proposition d'architecture pour répondre à la problématique précédemment décrite.

Pour illustrer notre propos, nous avons choisi un exemple dans le domaine linguistique, basé sur des services déployés[3]. Ici, nous voulons trouver la définition d'un mot en français sans avoir de dictionnaire français. Nous passons donc par une traduction vers l'anglais, par un dictionnaire anglais qui en donne la définition et enfin par le même traducteur pour obtenir la définition en français.

Comme précisé ci-dessus, la sélection des deux services, i.e. le dictionnaire et le traducteur, est considérée comme déjà effectuée. La première étape consiste à exprimer le but, l'état initial du système et la base de connaissances pour transformer le problème de composition en problème et domaine de planification distribuée. En sortie de planification, il faut transformer le plan en description exécutable de web service composite. Enfin, en fonction d'ontologies déterminant le procédé à suivre, la QoS décrite dans les services atomiques doit être agrégée pour calculer la QoS du service composite.

---

2. `http://core.imag.fr/`
3. Ces services sont décrits et hébergés sur `http://www.mindswap.org/2004/owl-s/services.shtml` .

Dans notre exemple, le but est de trouver une définition en français, l'état initial est que l'on a un mot en français. La base de connaissances permet d'exprimer que le traducteur peut traduire du français vers l'anglais et de l'anglais vers le français. Ces trois entrées sont définies sémantiquement[4] selon l'ontologie de données.

Ensuite, ces trois entrées et les descriptions des services sélectionnés pour la composition sont transformées en problème et domaine de planification. Puisqu'un planificateur distribué est utilisé, où chaque agent a des capacités précises, le domaine et la base de connaissances doivent être distribués. Nous avons choisi, à la manière de la Language Grid, de représenter chaque service abstrait par un agent, contenant donc les actions proposées par un seul service abstrait. Ceci a l'avantage de permettre à chaque organisation de mettre à jour sa base de connaissances sans prendre en compte les contraintes de ses partenaires. Enfin, les agents construisant coopérativement un plan, le but est commun.

Le plan généré par le planificateur n'étant pas directement exécutable, il faut le transformer en description de service composite. Il faut donc convertir, en formalisme OWL-S, le flux de contrôle, dans notre exemple une séquence de trois actions et le flux de données, représenté par les *bindings* de variables du plan. De plus, CoRe étant capable de générer un plan sous hypothèses, il faut transformer celles-ci sous forme de services abstraits. Par exemple, si le traducteur ne pouvait passer du français à l'anglais, CoRe pourrait formuler cette hypothèse qui serait traduite sous la forme d'un service abstrait prenant en entrée une phrase en français et en sortie une phrase en anglais.

Une fois le service composite généré, il reste à calculer la QoS résultante en fonction des QoS des services composés. Le calcul de paramètres non fonctionnels tels que le temps de réponse ou le coût se font de manière triviale (Michael C. Jaeger, 2004). Néanmoins, des paramètres spécifiques au domaine, tels que la qualité de traduction, ne sont pas calculables de façon directe. Pour cela, des ontologies définissant comment calculer chaque paramètre spécifique au domaine en fonction de chaque structure de contrôle sont nécessaires.

Conclusion et perspectives

En conclusion, notre architecture permet, contrairement aux systèmes existants, de s'adapter aux contraintes des environnements dynamiques dans lesquels les collaborations inter-organisationnelle évoluent. La procédure pour transformer un problème de composition en problème de planification distribuée a été décrite et validée par un démonstrateur se basant sur CoRe.

Néanmoins, il reste à valider cette proposition d'agrégation de QoS. De plus, CoRe étant encore en cours de développement, notamment au niveau de l'implémentation du dialogue multi-agents, il serait intéressant de continuer la validation de notre architecture au vu des nouvelles fonctionnalités qui seront implantées dans CoRe.

Finalement, la QoS pourrait être utilisée pour distribuer l'exécution de la composition. Un service ayant une influence critique sur la QoS globale de la composition

4. En RDF.

devrait être confiée à un agent d'exécution de confiance. De plus, les aspects multi-agents pourraient être renforcés en proposant, par exemple, de passer des contrats (Gateau *et al.*, 2005) pour vérifier si les contraintes de QoS sont respectées et proposer des pénalités en conséquence. La base de connaissance décrite pour l'instant manuellement pourrait aussi être générée automatiquement grâce à des agents d'information.

3. Bibliographie

ADETTI, D5.2 : Service Composition and Execution in IP2P, Technical report, CASCOM, August, 2006.

Bourdon J., *« Multi-agent systems for the automatic composition of semantic web services in dynamic environments »*, Master's thesis, Ecole Nationale Superieure des Mines de Saint-Etienne, 2007.

Gateau B., Khadraoui D., Dubois E., Boissier O., « Moise Inst : An Organizational Model for Specifying Rights and Duties of Autonomous Agents », *COORG 2005*, 2005.

Ghallab M., Nau D., Traverso P., *Automated Planning : theory and practise*, Morgan Kaufmann Publishers, 2004.

Hendler J., « Agents and the Semantic Web », *IEEE Intelligent Systems*, March, 2001.

Ishida T., « Language Grid : An Infrastructure for Intercultural Collaboration », *IEEE/IPSJ Symposium on Applications and the Internet (SAINT-06)*, p. 96-100, 2006.

Klusch M., Gerber A., Schmidt M., « Semantic Web Service Composition Planning with OWLS-Xplan », *Proceedings 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web, Arlington VA, USA*, 2005.

Martin D., Burstein M., Hobbs J., Lassila O., McDermott D., McIlraith S., Narayanan S., Paolucci M., Parsia B., Payne T., Sirin E., Srinivasan N., Sicara K., OWL-S : Semantic Markup for Web Services, Technical report, France Telecom, MINDL Maryland, NIST, Nokia, 2004.

Michael C. Jaeger Gregor Rojec-Goldmann G. M., « QoS Aggregation for Web Service Composition using Workflow Patterns », *Enterprise Distributed Object Computing Conference, Eighth IEEE International (EDOC'04)*, 2004.

Papazoglou M. P., Traverso P., Dustdar S., Leymann F., Krämer B. J., « 05462 Service-Oriented Computing : A Research Roadmap », *in* F. Cubera, B. J. Kr"amer, M. P. Papazoglou (eds), *Service Oriented Computing (SOC)*, n° 05462 in *Dagstuhl Seminar Proceedings*, Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2006.

Pellier D., Modèle dialectique pour la synthèse de plans, PhD thesis, IMAG - LEIBNIZ, December, 2005.

Singh M. P., Huhns M. N., *Service Oriented Computing : Semantics, Processes and Agents.*, John Wiley & Sons, 2005.

Traverso P., Pistore M., « Automated Composition of Semantic Web Services into Executable Processes », *International Semantic Web Conference (ISWC)*, 2004.

Virdell M., « Business processes and workflow in the Web services world », *developerWorks*, January, 2003.