Sujet de Master recherche 2010
Du bruit dans la forêt :
des arbres communicants pour vérifier et réaliser des Web-Services

Encadrants : Loïc Hélouët, Eric Badouel
Equipe : Distribcom, S4
mail: loic.helouet@irisa.fr, eric.badouel@irisa.fr
motc clés : Web Services, arbres, documents actifs.

1 Résumé

Les systèmes d'information évoluent de plus en plus vers des architectures de type "Web Services" [6], dans lesquelles des applications sont conçues par assemblage (on parle aussi d'orchestrations) de composants logiciels répartis sur le web. Ces composants doivent être coordonnées pour fournir des services de plus haut niveau. Les composants de base sur lesquels s'appuie une solution à base de services ne sont pas nécessairement conçus et maintenus par le créateur de l'application, et peuvent provenir de sites distincts. On peut même imaginer des orchestrations de services dans lesquelles des composants sont mis en compétition pour fournir la meilleurs réponse (la plus rapide ou la plus complète) à un utilisateur. Un concepteur de Web Services qui réutilise les solutions fournies par d'autres sites n'a pas a priori accès aux détails des implémentations et ne les perçoit que par ses interactions avec elles, c'est a dire en tant qu'environnement de son propre système. Les architectures de type Web Services sont donc par essence des systèmes *hétérogènes et ouverts*.

Dans ce contexte, un concepteur ne dispose pour les services qu'il invoque que de descriptions sommaires par des *interfaces*, qui indiquent comment appeler un service (adresse sur le web, mais aussi paramètres des appels) et qu'attendre en retour. Bien que les informations disponibles sur l'environnement soient assez parcellaires, il est important de pouvoir raisonner (par exemple vérifier des propriétés de sûreté) sur les comportements d'une orchestration, afin de donner des garanties aux utilisateurs du service que l'on définit de la sorte.

Le concept de *document actif*, initialement propose par Serge Abiteboul [1, 2] est un formalisme permettant d'implémenter et de raisonner sur les comportements de Web services. Un document actif est un document dont certaines parties ne sont pas explicitées (ce sont simplement des références à d'autres services), mais qui peuvent être évaluées par un appel aux services référencés (on peut se représenter cette situation comme un document html contenant des références à d'autres pages web). Les documents actifs permettent de concevoir des services de haut niveau, mais surtout, rendent possible des évaluation "paresseuses" de données : plutôt que de renvoyer un énorme document dont une large partie n'est pas utilisée dans 90% des cas, un service peut retourner un document contenant les informations les plus utiles, et un lien vers un service à invoquer pour obtenir le reste.

Parmi les modèles proposés pour les documents actifs, on peut citer AXML [3], TPRS [5]. Les principes généraux des deux langages sont assez similaires : une architecture Web service est décomposée en sites. Les sites possèdent des données et des services, invocables localement ou par les autres sites. Les réponses aux invocations de services et les ajouts de données sont faits à l'aide de requêtes (queries). L'évaluation de la valeur à retourner par un service peut necessiter l'appel récursif d'un ou plusieurs sous-services. A partir de ces principes simples, on peut créer autant de variantes de ces modèles qu'il y a de langages

de requêtes sans en changer fondamentalement l'esprit. L'implémentation réelle d'un tel système consiste tout simplement à animer un modèle, c'est à dire lancer localement des requêtes sur une base de données, et assurer la communication entre sites. Si l'on suppose que les communications entre sites se font sans pertes (une invocation ne peut pas être perdue, ni son retour), il n'y a donc que peu de différences entre les comportements d'un modèle et l'implémentation de celui-ci.

Dans ce stage nous proposons l'étude d'un formalisme proche d'AXML, les *arbres communicants*. Les principes de base de ce langage sont :

- les données sont représentées sous forme d'arbres (à la manière de ce qui se fait pour XML) contenant des tags, des données, et des invocations de services.
- Les modifications/interrogations documents se font à l'aide de *transductions d'arbres* [4] (i.e. une fonction qui prend un arbre en entrée et retourne un autre arbre)
- Les appels aux services d'un autre site et leurs retours se font de manière asynchrone, mais non FIFO.
- les services externes non spécifiés peuvent être décrits par des interfaces, elle même spécifiées comme des transductions d'arbres.

A partir de ce modèle, on essaiera de fournir des algorithmes pour répondre aux questions suivantes:

- (**Accessibilité :**) Une configuration du système contenant un motif particulier (ou pattern) dans l'un de ses arbres est elle atteignable à partir d'une configuration de départ ?
- (**Terminaison :**) Un appel à un service se termine-t-il toujours ?

Il est tout à fait possible que les problèmes ci-dessus (et bien d'autres) se révèlent indécidables. On cherchera alors des restrictions suffisantes sur les transductions utilisées comme requêtes pour rendre l'accessibilité et la terminaison décidables.

2 Compétences requises

Aucune a priori ! L'introduction des concepts et premiers modèles nécessaires à la bonne compréhension du sujet sera assurée par les encadrants. Il est toutefois préférable que le candidat soit un minimum attiré par l'algorithmique et les méthodes formelles.

References

[1] S. Abiteboul, O. Benjelloun, I. Manolescu, T. Milo, and R. Weber. Active XML: A data-centric perspective on web services. In *BDA*, 2002.

[2] S. Abiteboul, O. Benjelloun, and T. Milo. Positive active XML. In *PODS*, pages 35–45, 2004.

[3] S. Abiteboul, L. Segoufin, and V. Vianu. Static analysis of active XML systems. In *Symposium on Principles of Database Systems (PODS)*, pages 221–230, 2008.

[4] H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: http://www.grappa.univ-lille3.fr/tata, 2007. release October, 12th 2007.

[5] Blaise Genest, Anca Muscholl, Olivier Serre, and Marc Zeitoun. Tree pattern rewriting systems. In *ATVA*, pages 332–346, 2008.

[6] Ed Ort. Service-oriented architecture and web services: Concepts, technologies, and tools. Technical report, SUN, 2005. http://java.sun.com/developer/technicalArticles/WebServices/soa2/soa2.pdf.