Titre : *Cadre formel pour la spécification et la vérification de flots de communication de processus distribués dans le Cloud*

Encadrants : Christophe Gaston (CEA-LIST) Pascale Le Gall (CentraleSupélec)

Localisation : Nano-INNOV - CEA-LIST - Université Paris-Saclay – laboratoire LECS

Les clouds sont constitués de serveurs interconnectés via internet, sur lesquels on peut implémenter des systèmes faisant usages d'applications et de bases de données déployées sur les serveurs. L'informatique basée sur les clouds gagne considérablement en popularité, y compris pour y déployer des systèmes critiques. De ce fait, disposer d'un cadre formel pour raisonner sur ce type de systèmes devient une nécessité. Une exigence sur un tel cadre est qu'ils permettent de raisonner sur les concepts manipulés dans un cloud, ce qui inclue naturellement la capacité à raisonner sur des systèmes distribués, composés de sous-systèmes déployés sur différentes machines et interagissant par passage de messages pour réaliser des services. Dans ce contexte, la facilité à raisonner sur les flots de communications est un élément central.

Modéliser des communications asynchrones entre des processus concurrents est possible dans une variété de formalismes, tels que les algèbres de processus [1], les réseaux de Petri [2], les automates distribués [3], ou bien encore les formalismes dérivés de Message Sequence Charts (MSC) [4]. Les MSC sont des modèles graphiques représentant des échanges d'informations entre des sous-systèmes. Diverses ramifications des MSCs, dont les diagrammes de séquence UML (UML-SD) [5], ont été proposés et nous appelons les langages de cette famille "langages d'interaction" (LI). Les interactions sont intéressantes en raison de leur nature graphique et de leur facilité de compréhension. Les LI permettent de décrire des scénarios de communication de manière claire et intuitives: a) un trait vertical, appelé ligne de vie, représente la dynamique de chaque sous-système en décrivant de haut en bas la succession des événements tels que perçus par le sous-système, b) les échanges de messages sont représentés par des flèches horizontales reliant deux lignes de vie et décorées par des messages c) des opérateurs de haut niveau tels que divers types de séquencements, la composition parallèle, le choix, la répétition pour structurer des scénarios simples, sont disponibles.

Pour pouvoir utiliser un LI dans le cadre d'une approche formelle, il faut le munir d'une sémantique formelle. Beaucoup de travaux ont été menés en ce sens [6]. Nous proposons de nous baser sur des travaux récents [7] très complets puisqu'ils associent un LI avec une sémantique dénotationnelle et une sémantique opérationnelle tout en prouvant leur équivalence (en Coq). La sémantique dénotationnelle est un outil très efficace pour raisonner sur les interactions elles-mêmes (pour prouver l'équivalence de deux interactions par exemples, ou pour définir des formes normales) alors que la sémantique opérationnelle est naturellement adaptée pour prouver la correction d'algorithmes (comme par exemple, des algorithmes de vérification à l'exécution [8,9]). Ces travaux proposent de plus un cadre outillé pour l'exploration sémantique et la vérification à l'exécution, dédié au interactions (outil IAT - Interaction Analysis Tool).

Nous proposons d'étendre les travaux précédemment mentionnés en définissant un cadre formel de spécification, basé sur les interactions et dédié à la spécification de systèmes déployés sur des clouds. La piste envisagée est d'inscrire le LI précédemment mentionné dans la cadre de la théorie des institutions comme cela a été fait dans d'autres travaux [10] pour un LI plus restreint que celui visé ici. Ce dernier sera étendu pour augmenter son expressivité et permettre de sous-spécifier les comportements d'un système, pour se concentrer sur les aspects

critiques en faisant abstraction des autres. Des opérateurs seront introduits pour refléter les activités de génie logiciel inhérent à l'informatique basée sur les clouds. Un cadre pour la vérification de propriété de sûreté et/ou de sécurité sur des modèles d'interaction sera étudié. Ce cadre sera étendu pour pouvoir supporter des activités de génie logiciel mettant en jeux plusieurs équipes, comme c'est le cas en pratique pour les systèmes visés. Cela inclura la définition d'opérateur de composition et de raffinement. Des résultats de préservation de propriété au travers de ces opérateurs seront étudiés.

[1] Rensink, A., Wehrheim, H.: Weak sequential composition in process algebras. In: Jonsson, B., Parrow, J. (eds.) 5th Conf. on Concurrency Theory (CONCUR). pp. 226–241. Lecture Notes in Computer Science, Springer (1994).
[2] Haddad, S., Khmelnitsky, I.: Dynamic recursive petri nets. In: Janicki, R., Sidorova, N., Chatain, T. (eds.) Application and Theory of Petri Nets and Concurrency. pp. 345–366. Springer International Publishing, Cham (2020).
[3] Akshay, S., Bollig, B., Gastin, P., Mukund, M., Narayan Kumar, K.: Distributed timed automata with independently evolving clocks. In: van Breugel, F., Chechik, M. (eds.) 19th Conf. on Concurrency Theory (CONCUR). pp. 82–97. Springer Berlin Heidelberg, Berlin, Heidelberg (2008).
[4] Mauw, S., Reniers, M.A.: Operational semantics for msc'96. Computer Networks 31(17), 1785–1799 (1999).
[5] OMG: Unified Modeling Language v2.5.1. omg.org/spec/UML/2.5.1/PDF (12 2017).
[6] Micskei, Z., Waeselynck, H.: The many meanings of uml 2 sequence diagrams: a survey. Software & Systems Modeling 10(4), 489–514 (2011).
[7] Mahe, E., Gaston, C., Le Gall, P.: Equivalence of Denotational and Operational Semantics for Interaction Languages. In: Ameur, Y., Craciun, F. (eds.) 16th International Symposium on Theoretical Aspects of Software Engineering (TASE). pp. 113-130. Lecture Notes in Computer Science, Springer (2022).
[8] Mahe, E., Gaston, C., Le Gall, P.: Revisiting semantics of interactions for trace validity analysis. In: Wehrheim, H., Cabot, J. (eds.) Fundamental Approaches to Software Engineering (FASE). pp. 482–501. Springer International Publishing, Cham (2020)
[9] Mahe, E., Bannour, B., Gaston, C., Lapitre, A., Le Gall, P.: A small-step approach to multi-trace checking against interactions. 36th Annual ACM Symposium on Applied Computing (SAC). pp. 1815–1822. Association for Computing Machinery (2021).
[10] Knapp, A., Mossakowski, T.: UML Interactions Meet State Machines - An Institutional Approach. In: 7th Conf. on Algebra and Coalgebra in Computer Science (CALCO). Leibniz International Proceedings in Informatics (LIPIcs), vol. 72 (2017).