

Techniques d'Apprentissage Automatique Appliquées Au Génie Logiciel : Application à la migration de Spring MVC vers Spring Microservices (JAVA)

Stage proposé en collaboration entre le laboratoire LIRMM (<http://www.lirmm.fr/>) et l'entreprise Berger Levraut (<https://www.berger-levraut.com/fr/>)

Encadrants :

- Abdelhak-Djamel Seriai, MCF Dept Info, FDS, UM, <http://www.lirmm.fr/~seriai/>, seriai@lirmm.fr
- Anas Shatnawi, ingénieur de recherche chez Berger Levraut (BL), <https://www.berger-levraut.com/fr/>
- Pascal Zaragoza, Doctorant Chez BL, <https://www.berger-levraut.com/fr/>

Contexte

Le style architectural Microservices connaît aujourd'hui un engouement sans précédent en raison de ses caractéristiques intrinsèques. Une architecture à base de Microservices est composée d'un ensemble de services modulables, relativement indépendants, centrés autour d'objectifs métier différents, qui assurent une gestion décentralisée des données et qui prennent en compte les aspects liés à la (re)configuration pour s'adapter à des besoins et des contextes variables. Idéalement, chaque micro-service s'exécute dans son propre processus et communique à travers un mécanisme léger et bien défini pour atteindre un objectif métier qui lui est propre. Ainsi, au lieu de redéployer toute l'application quand elle est conçue comme un monolithe, les services d'une application conçue comme un ensemble de Microservices peuvent être déployés, ajustés et redéployés indépendamment sans compromettre l'intégrité de l'application en question.

Le style architectural Microservices est une solution aux problèmes connus du style monolithique qui sont : la difficulté de passage à l'échelle, la difficulté à être déployées efficacement, la faible tolérance aux pannes, la difficulté d'adoption des bonnes pratiques de DevOps (continuité de l'intégration, de la livraison et du déploiement), etc.

Spring est un Framework open source pour construire et définir l'infrastructure d'une application Java. Le Framework a une architecture de type MVC (Model-View-Controller) et ses composants servent pour développer des applications Web flexibles et faiblement couplées. Le modèle MVC permet de séparer les différentes parties d'une application web à savoir la gestion de requêtes d'entrée envoyées par le client, la logique métier et logique UI (affichage résultats en réponses aux requêtes) tout en assurant un couplage moins fort (Lazy) entre les différentes classes de l'application. Le but de Spring est de faciliter et de rendre productif le développement d'applications, particulièrement les applications d'entreprises.

« L'apprentissage automatique (en anglais : machine learning) est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches mathématiques et statistiques pour donner aux ordinateurs la capacité d'« apprendre » à partir de données. L'apprentissage automatique comporte généralement deux phases. La première consiste à estimer un modèle à partir de données, appelées observations, qui sont disponibles et en nombre fini, lors de la phase de conception du système. La seconde phase correspond à la mise en production : le modèle étant déterminé, de nouvelles données peuvent alors être soumises afin d'obtenir le résultat correspondant à la tâche souhaitée ».

Objectif et étapes

L'objectif de ce stage est de réaliser une partie d'un outil qui permet d'implémenter une approche existante de migration d'un code source du style monolithique vers le style Microservices en s'appuyant, principalement, sur des techniques d'apprentissage automatique (Clustering, Génétique, Q-learning). Ce projet de TER M1 est structuré en 4 étapes :

- 1) Implémenter deux algorithmes existants proposés par l'équipe MaREL du laboratoire Lirmm qui permettent d'identifier des clusters de Microservices à partir de l'analyse du code source de la partie Backend d'une application web développée en Spring. Les deux algorithmes concernent un algorithme de clustering hiérarchique et un algorithme génétique mono-objectif.
- 2) Proposer une approche basée sur le Q-learning pour l'identification des MicroServices.
- 3) Implémenter quelques patrons de refactoring existants proposés par l'équipe MaREL pour pouvoir créer des Microservices à partir du code source d'une application monolithique.
- 4) Implémenter une interface IHM Web (SaaS : Software as Service) pour pouvoir offrir à un architecte les deux services précédents (identifier des clusters de Microservices et la création des Microservices à partir de ces clusters).

Pourquoi choisir ce projet :

- Acquérir des compétences certaines en développement logiciel (et architecture logicielle), principalement, des frameworks de développement tel que Spring.
- Maîtrise de plusieurs outils de compréhension de code java tels que Spoon ou Java JDT. Ces outils sont très utilisés dans le monde des entreprises pour l'étape d'évolution et de maintenance de logiciels.
- Comprendre un des enjeux majeurs dans le monde du développement logiciel, à savoir, l'évolution de logiciels existants et leur modernisation.
- Découvrir plusieurs thématiques dans le monde du développement logiciel : la rétro ingénierie (reverse engineering), la réingénierie (reengineering), l'analyse du code source, les algorithmes génétiques, etc.
- Apprendre des techniques d'apprentissage automatique appliquées au domaine du génie logiciel.
- Avoir une collaboration avec le monde des entreprises (l'entreprise BL, un des grands éditeurs logiciels en France).
- Etc.

Si vous avez des questions, contactez Abdelhak-Djamel Seriai
seriai@lirmm.fr

Références Bibliographiques

- 1) Anfel Selmadji: From monolithic architectural style to microservice one: structure-based and task-based approaches. (Du style architectural monolithique vers le style microservice: approches basées sur la structure et sur les tâches). University of Montpellier, France, 2019
- 2) Anfel Selmadji, Abdelhak-Djamel Seriai, Hinde-Lilia Bouziane, Christophe Dony, Rahina Oumarou Mahamane: Re-architecting OO Software into Microservices - A Quality-Centred Approach. ESOC 2018: 65-73
- 3) Zakarea Alshara, Abdelhak-Djamel Seriai, Chouki Tibermacine, Hinde-Lilia Bouziane, Christophe Dony, Anas Shatnawi: Materializing Architecture Recovered from Object-Oriented Source Code in Component-Based Languages. ECSA 2016: 309-325
- 4) Microservice Patterns: With examples in Java . Chris Richardson, Aiden Humphreys, Lou Fernandez, Manning Publications. Novembre 2018. Spring par la pratique. Arnaud Cogoluègues, Thierry Templier, Julien Dubois, Jean-Philippe Retaillé. Juillet 2009. Editeur Eyrolles.
- 5) Spoon - Source Code Analysis and Transformation for Java : <http://spoon.gforge.inria.fr/>
- 6) AST : https://www.eclipse.org/articles/article.php?file=Article-JavaCodeManipulation_AST/index.html
- 7) Parseur AST dans JDT : <http://www.programcreek.com/2011/01/a-complete-standalone-example-of-astparser/>
- 8) AST View : <http://www.eclipse.org/jdt/ui/astview/index.php>
- 9) Patron visiteur : [https://fr.wikipedia.org/wiki/Visiteur_\(patron_de_conception\)](https://fr.wikipedia.org/wiki/Visiteur_(patron_de_conception))
- 10) Tutoriel Spring : introduction Spring MVC : <http://objjs.com/tutoriel-spring-n9-introduction-spring-mvc/>